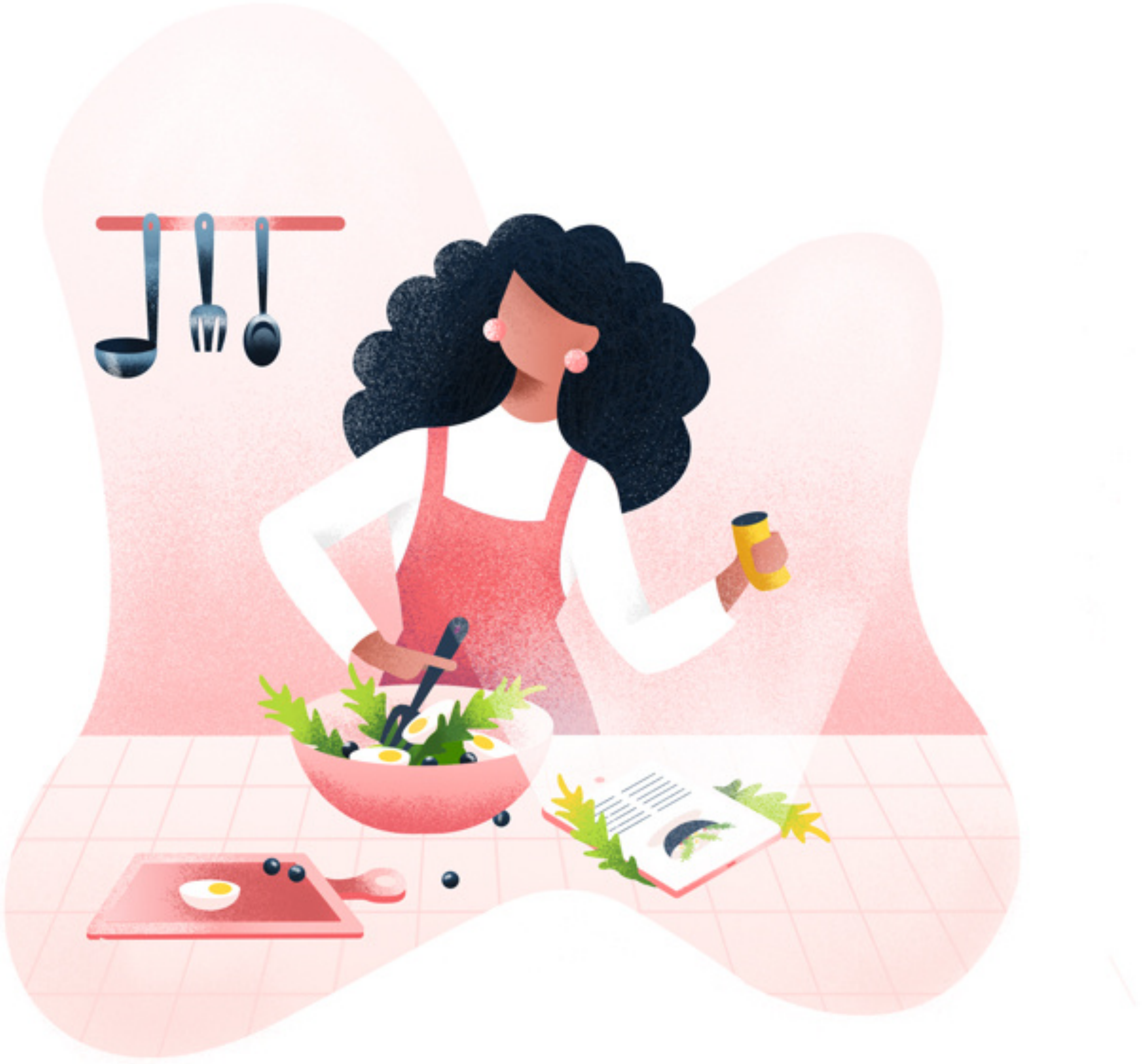


What's cooking?

Predicting cuisine
by provided ingredients
& market basket analysis

Gabriela Dvořáková, Michal Lehončák

[image source]

Data mining agenda

- **Part 01**: Goal definition
- **Part 02**: Dataset
- **Part 03**: Preprocessing
- **Part 04**: Prediction
- **Part 05**: Results discussion
- **Part 06**: Market Basket Analysis

Part 01

What’s the goal?

Predict the category of a dish's cuisine given a list of its ingredients.

INGREDIENTS

romaine lettuce
black olives
grape tomatoes
garlic
pepper
purple onion
seasoning
garbanzo beans
feta cheese crumbles

→

CUISINE
greek

[image source]

Part 02

Dataset

Kaggle competition free dataset with 3 features: cuisine, recipe id, ingredients

cuisine	id	ingredients
greek	10259	['romaine lettuce', 'black olives', 'grape tomatoes', 'garlic', 'pepper', 'purple onion', 'seasoning', 'garbanzo beans', 'feta cheese crumbles']
southern_us	25693	['plain flour', 'ground pepper', 'salt', 'tomatoes', 'ground black pepper', 'thyme', 'eggs', 'green tomatoes', 'yellow corn meal', 'milk', 'vegetable oil']
filipino	20130	['eggs', 'pepper', 'salt', 'mayonaise', 'cooking oil', 'green chilies', 'grilled chicken breasts', 'garlic powder', 'yellow onion', 'soy sauce', 'butter', 'chicken livers']
indian	22213	['water', 'vegetable oil', 'wheat', 'salt']

TRAINING SET

39774

recipes

TESTING SET

9944

recipes

Part 02

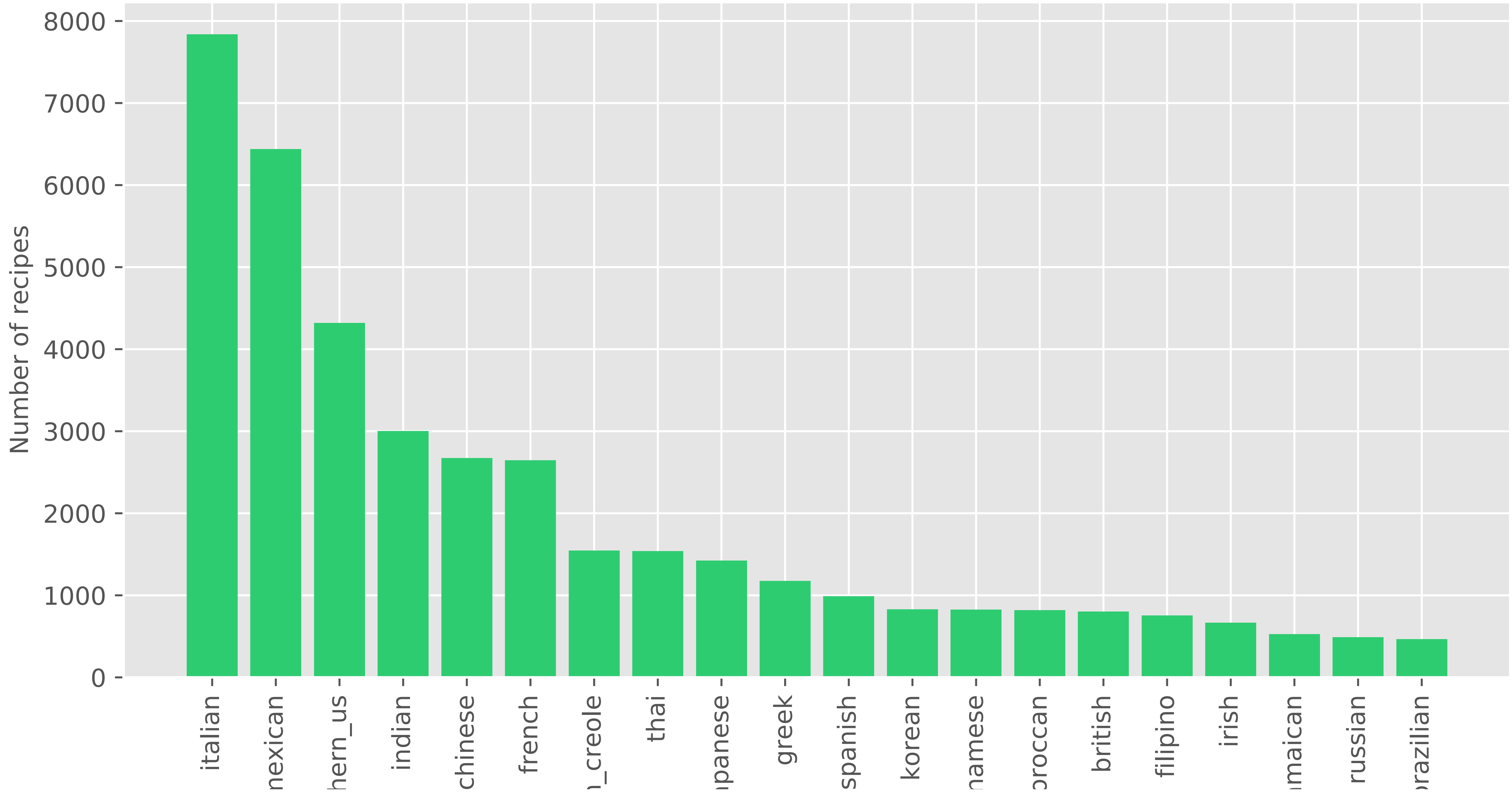
Dataset

CLASSES

20

cuisines

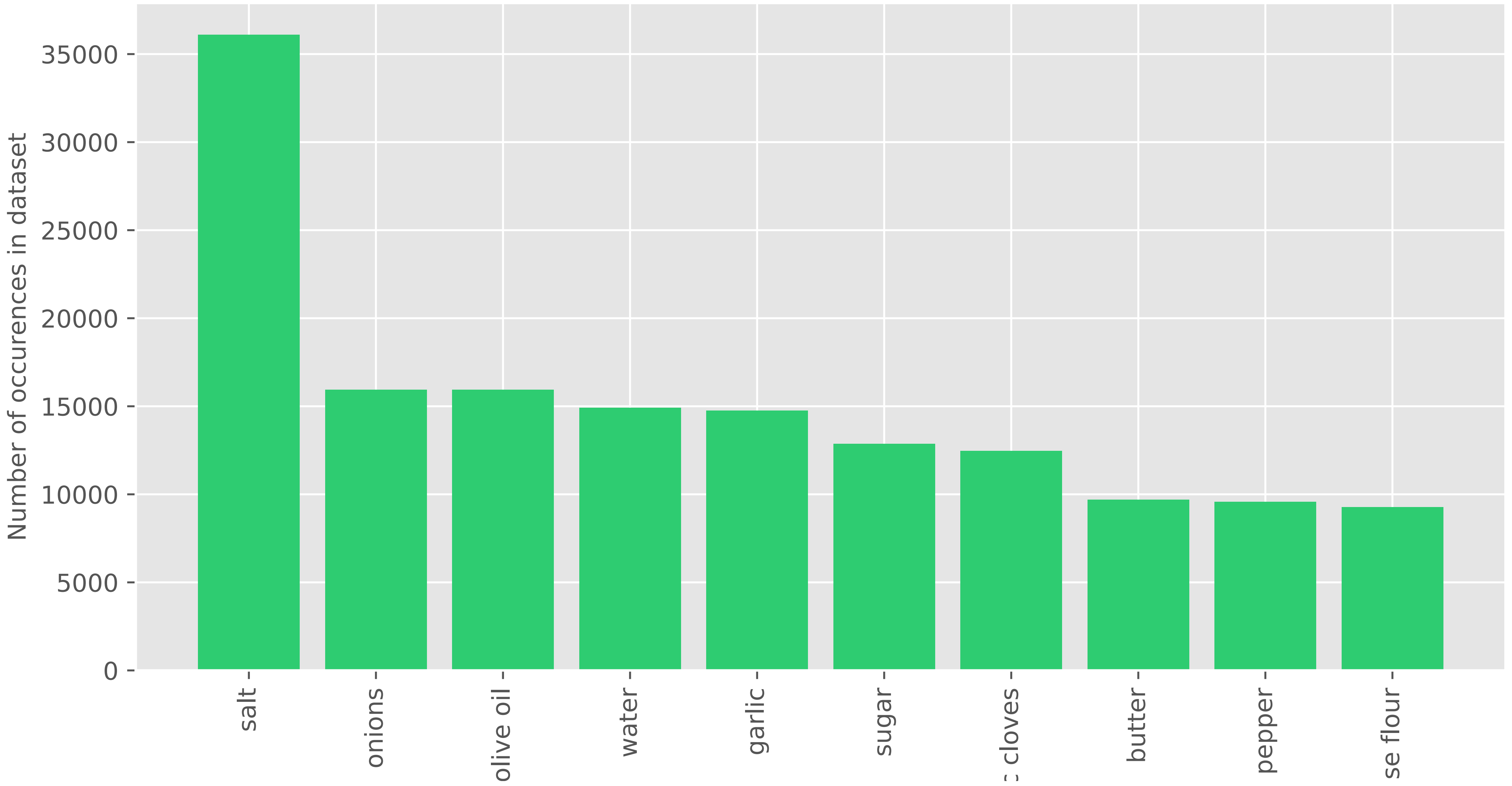
- number of dishes per cuisine:

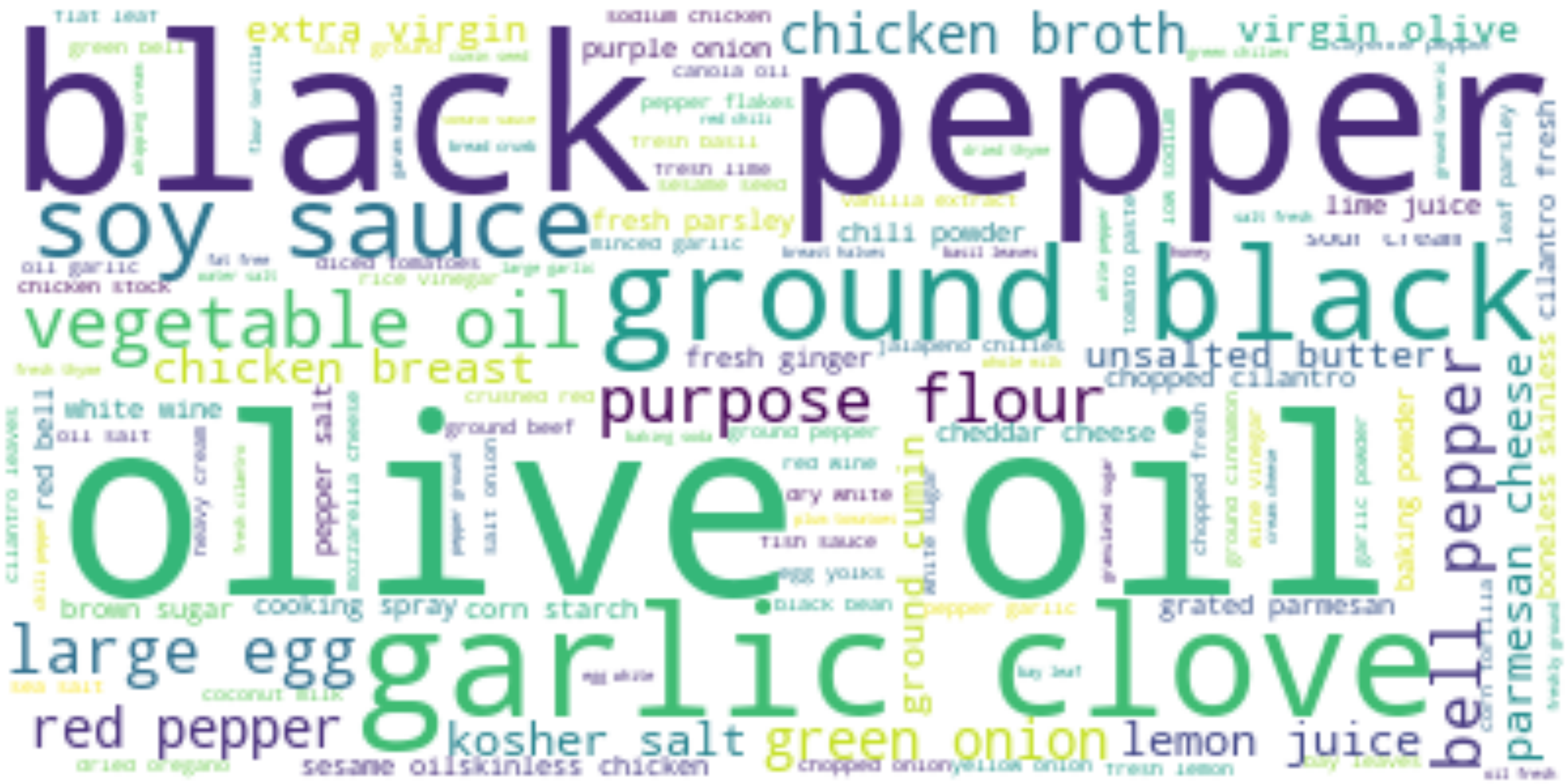


Part 02

Dataset

- most frequent ingredients:





Part 03

Preprocessing

1. Ingredient concatenation

- many ingredients have multiple names:

’finely chopped onion’ & ‘diced onions’ & ‘onions’

- computer views them as two distinct ingredients
- to solve this problem, ingredients are:
    - **concatenated** to form a single string
    - **tokenized**

example:

['water', 'vegetable oil', 'wheat', ‘salt’]

↓

[‘water vegetable oil wheat salt']

2. Normalizing Case

- all words are converted to lowercase

example:

‘KRAFT Zesty Italian Dressing’

↓

‘kraft zesty italian dressing'

Part 03

Preprocessing

3. Punctation removal

- removing any redundant symbols from the set:

'!"#$%&'()*+,-./:;<=>?@[]^_`{|}~™®'

4. Digits removal

- numbers do not hold much informational value for cuisine prediction

example:

‘1% low-fat milk’

’40% less sodium taco seasoning’

Part 03

Preprocessing

5. Porter Stemming

- ’tomatoes’ & ‘tomato’ are the same ingredient
- to achieve that, we use **stemming**`:

process of reducing words to their word stem, base or root form by removing characters from the word endings

example:

['tomatoes', 'sweetened, 'onions']

↓

[‘tomato’, ‘sweeten’, ‘onion’]

6. Rare words removal

- words that appear **less then 3 times** are removed from the ingredients, as their occurence does not provide enough prove to tell it discriminates one dish from another

examples of rare words (after stemming):

['poupon', 'krachai', 'nusalt', 'bluefish', 'rapini', 'rouget', 'shanghaistyl', 'moscato', 'dasti', 'hors', 'delux', 'silk', 'cupcak', 'surimi', 'dream', ‘hint’…]

Part 03

Preprocessing

Features encoding: Binary representation

each dish is represented by a **vector** of length of the number of unique ingredients

feature vector at i-th term may be 1 or 0

1: i-th ingredient appears in the example

0: otherwise

ingredient cannot appear more than once

id	‘olive’	‘garlic’	‘pepper’	‘milk’	‘broccoli’	...
10259	1	1	1	0	0	...
25693	0	0	1	0	0	...
20130	0	0	0	0	0	...
22213	0	0	0	1	0	...

Part 04

Prediction

- we compared 5 different predictive models

- models are evaluated on the test-train 20%-80% split of the training dataset

1 Logistic Regression

2 Gaussian Bayes classifier

3 Random Forest classifier

4 KNN classifier

5 Neural Network

Part 04

1. Logistic Regression

TRAIN ACCURACY

84.11%

TEST ACCURACY

78.66%

Part 04

2. Gaussian Bayes classifier

Assumption: conditional independence assumption

we assume the probability of one ingredient does not depend on the presence of different ingredient in a dish

$$p(ingr_i \mid ingr_{i+1}, \ldots, ingr_n, Dish_k) = p(ingr_i \mid Dish_k)$$

TRAIN ACCURACY

29.61%

TEST ACCURACY

23.77%

Part 04

3. Random Forest classifier

TRAIN ACCURACY

99.97%

TEST ACCURACY

75.80%

Part 04

4. K nearest neighbors

Optimal K: 15

number of nearest neighbors ‘K’ found using Grid Search

Metric: minkowski

TRAIN ACCURACY

70.13%

TEST ACCURACY

65.48%

Part 04

5. Neural Network

- we acquired the best results with the following architecture:

EPOCHS

100

BATCH SIZE

64

Layer (type)	Output shape	Param #
Dense	(None, 1024)	2749440
Batch Normalization	(None, 1024)	4096
Dropout	(None, 1024)	0
Dense	(None, 512)	524800
Dropout	(None, 512)	0
Dense	(None, 256)	131328
Dropout	(None, 256)	0
Dense	(None, 20)	5140

TRAIN ACCURACY

99.93%

TEST ACCURACY

80.38%

Part 05

Results discussion

- using the 5 classification algorithms discussed,
  the best performance observed is from the NN model
- we created a final model by training NN on the **whole training dataset**
- submittion to kaggle we achieved the following accuracy:

FINAL SCORE ON KAGGLE

80.03%

Part 05

Results discussion

- we compared our results with the kernel cuisine-classification, which achieved the best accuracy of 78.88% using the OVA SVM algorithm
- different feature encoding method - TF-IDF
- less thorough preprocessing steps (no sparse words removal)
- our model performed better by almost 2 % using the deep neural network

Thank you for your attention.

References:

[0] https://www.kaggle.com/c/whats-cooking

[image source]