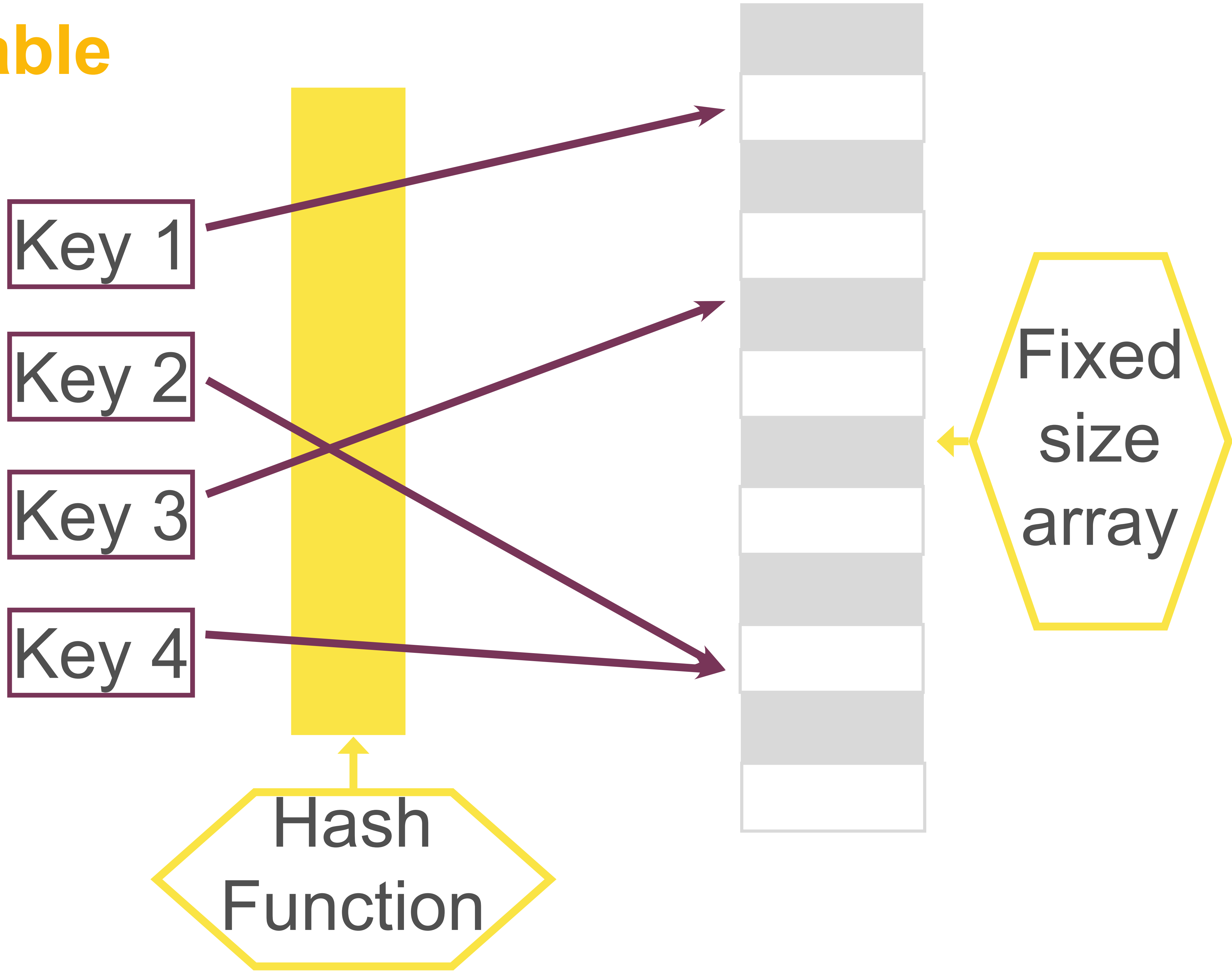
Feature Engineering for Texts

Part 2

Hashing trick

Hash Table



Key 1

Key 2

Key 3

Key 4

Hash Function

Fixed size array

Hash function f(s):

f(“this”) = 2

f(“apple”) = 10

f(“fruit”) = 5

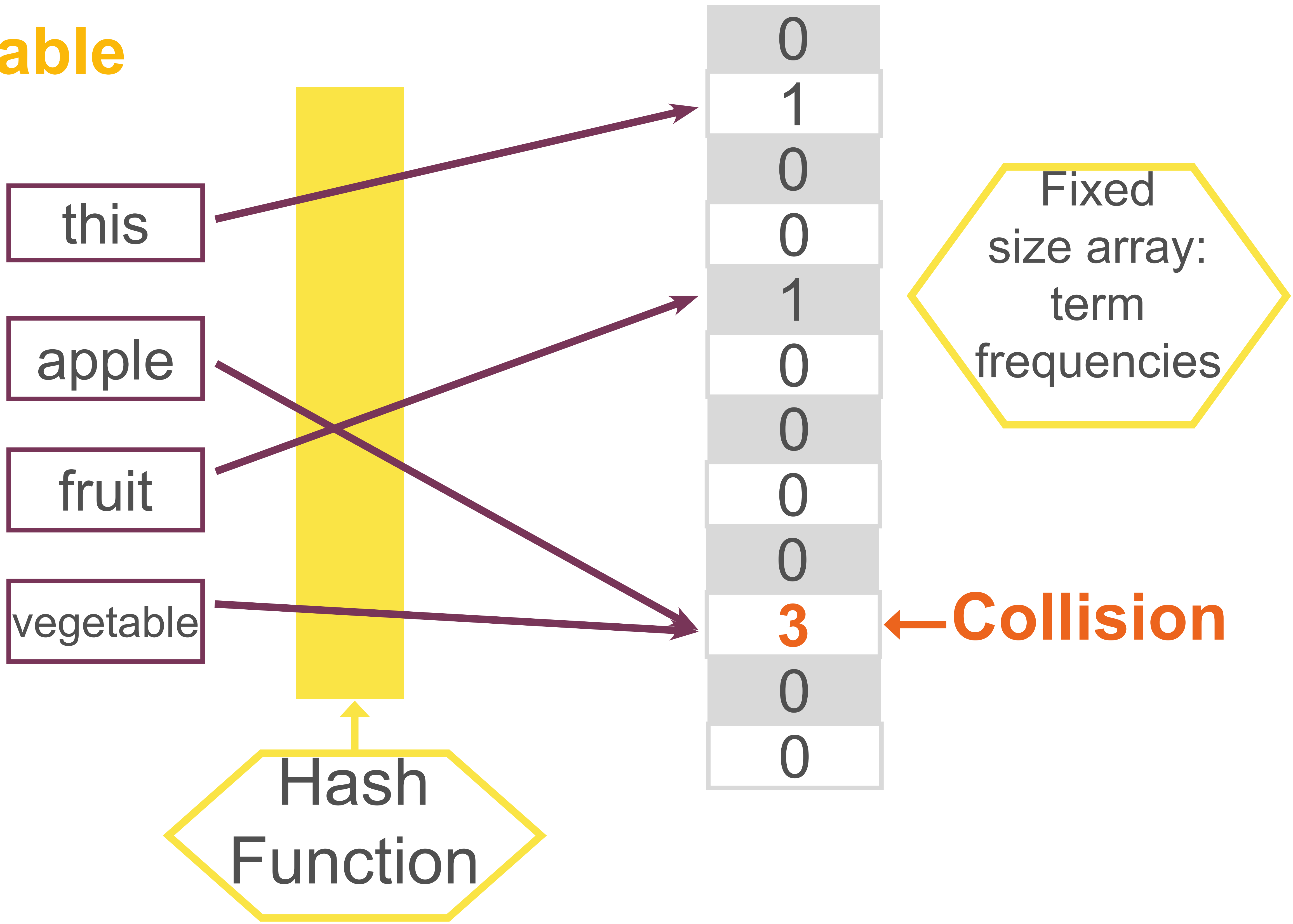
…

Common hash functions:

- MurMur3 hash
- Jenkins hash
- CityHash
- md5 hash

Hashing trick

Hash Table



this

apple

fruit

vegetable

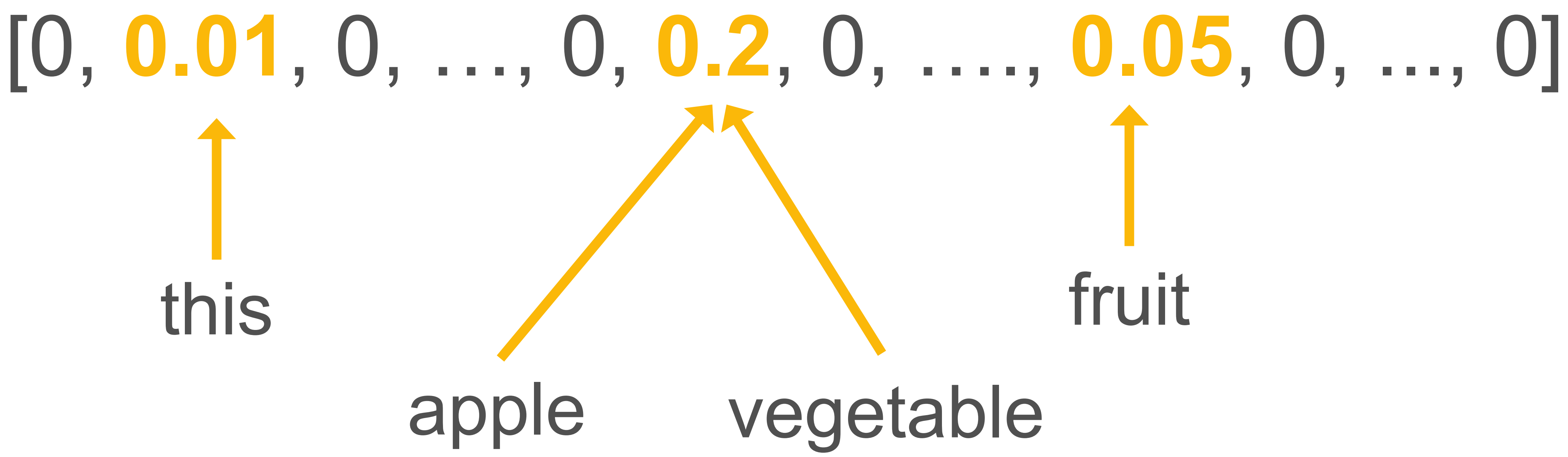
0
1
0
0
1
0
0
0
0
3
0
0

Fixed size array: term frequencies

Collision

Hash Function

How to find an index of a word in the feature vector?



[0, **0.01**, 0, …, 0, **0.2**, 0, …., **0.05**, 0, ..., 0]

this

apple

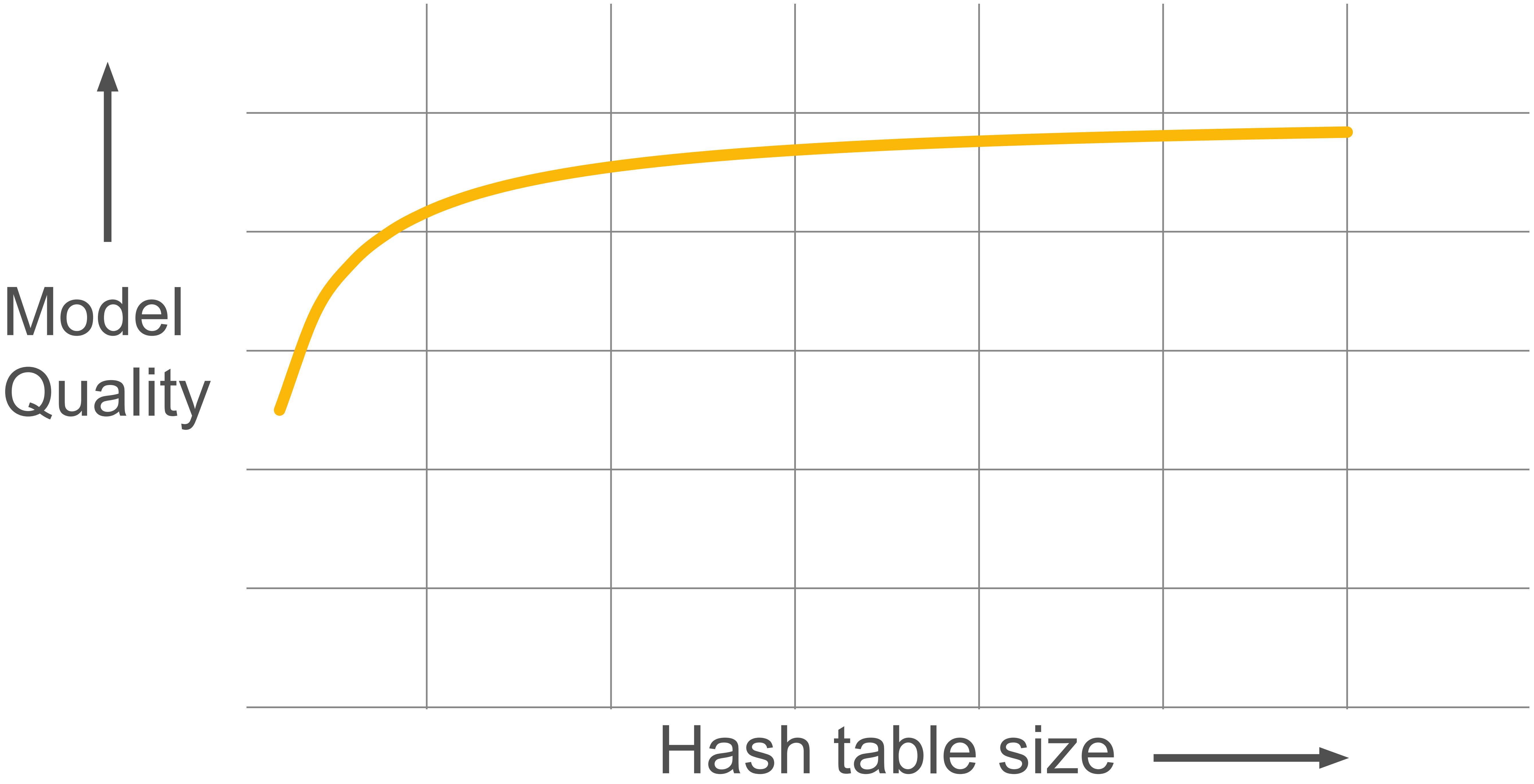
vegetable

fruit

Dictionary: **lookup** in dictionary

Hashing trick: calculate the **hash function** of the word

Hash table size vs. Model quality



Dictionary vs. Hashing trick

Dictionary	Hashing trick
No collisions	**Collisions**
Need to store dictionary for learning and in production	No dictionary Calculations on-the-fly
Slow if dictionary is large – $O(\log \lvert D \rvert)$	Fast – $O(1)$
Feature vector size = unique words and n-grams count **Variable memory footprint**	Feature vector size = size of the hash table (fixed) Fixed memory footprint