Universität Augsburg
Fakultät für Angewandte Informatik

03 Language Modeling and Machine Learning Basics

Introduction to Natural Language Processing

Prof. Dr. Annemarie Friedrich

Summer Semester 2025

Learning Goals

- Explain n-gram language modeling, compute probability of a sequence according to a bigram language model
- Compute perplexity for a test corpus
- Become familiar with basic machine learning terminology and concepts

N-gram Language Modeling

N-Gram Language Modeling

The dog chased the ?

History $P(w_1,w_2,w_3,w_4)$

$P(w_5|w_1,w_2,w_3,w_4)$

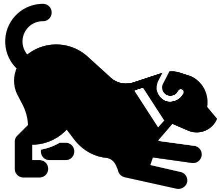
Probability of next word

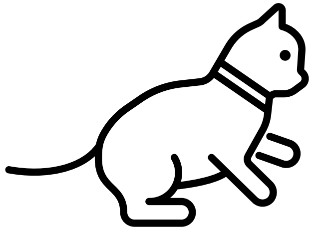
The dog chased the mouse.

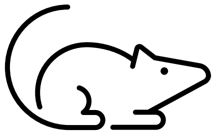
The dog chased the cat.

$P(W) = P(w_1,w_2,w_3,w_4,w_5\ldots w_n)$

Probability of sentence / sequence.





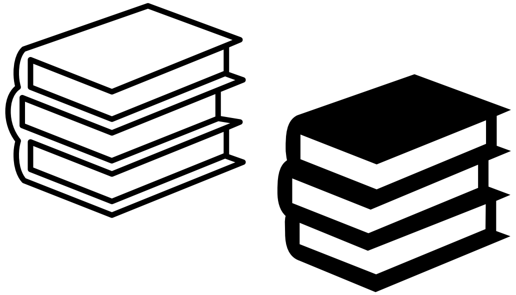


Assumption

Access to large amounts of machine-readable text data (corpora).

Common Crawl

WIKINEWS



das Korpus

der Korpus

Google Books

Likeliness of sentences occurring in these text corpora ~ probability of sentences in real-world.

N-Gram Language Modeling

Need to estimate these!

$P(w_5|w_1,w_2,w_3,w_4)$
Probability of next word

$P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$

Probability of sentence / sequence.

Chain Rule of Probability: **P(A,B) = P(A)P(B|A)**

$P(w_1,w_2,w_3,w_4,w_5 \ldots w_n) = P(w_1)*P(w_2|w_1)*P(w_3|w_1,w_2)$

$* P(w_4 | w_1,w_2,w_3)* P(w_5 | w_1,w_2,w_3,w_4)* \ldots * P(w_n | w_1,w_2,w_3,w_4,\ldots,w_{n-1})$

N-Gram Language Modeling

$P(w_5|w_1,w_2,w_3,w_4)$
Probability of next word

Belgium became the first 2024 Eurovision Song Contest participant country to ______

Just count in large text corpus?

$$P(\text{announce}|\text{Belgium became the first 2024 Eurovision Song Contest participant country to})$$

$$\frac{\text{count(Belgium became the first 2024 Eurovision Song Contest participant country to announce)}}{\text{count(Belgium became the first 2024 Eurovision Song Contest participant country to)}}$$

Why does this not work?

Markov Assumption

P(announce|Belgium became the first 2024 Eurovision Song Contest participant country to)

≈ P(announce | to) bigrams

or

≈ P(announce | country to) trigrams

or

≈ P(announce | participant country to) four-grams

Andrei Markov
(1856-1922)

N-Gram Language Modeling

$P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$

Probability of sentence / sequence.

Problem: what if all of these are high, but one probability score is 0?

Chain Rule of Probability:

$$P(w_1,w_2,w_3,w_4,w_5 \ldots w_n) = P(w_1)*P(w_2|w_1)*P(w_3|w_1,w_2)*P(w_4|w_1,w_2,w_3) * P(w_5|w_1,w_2,w_3,w_4)* \ldots * P(w_n|w_1,w_2,w_3,w_4,\ldots,w_{n-1})$$

Unigram language model: $P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n) \approx \prod_i P(w_i)$

Bigram language model: $P(W) \approx \prod_i P(w_i|w_{i-1})$

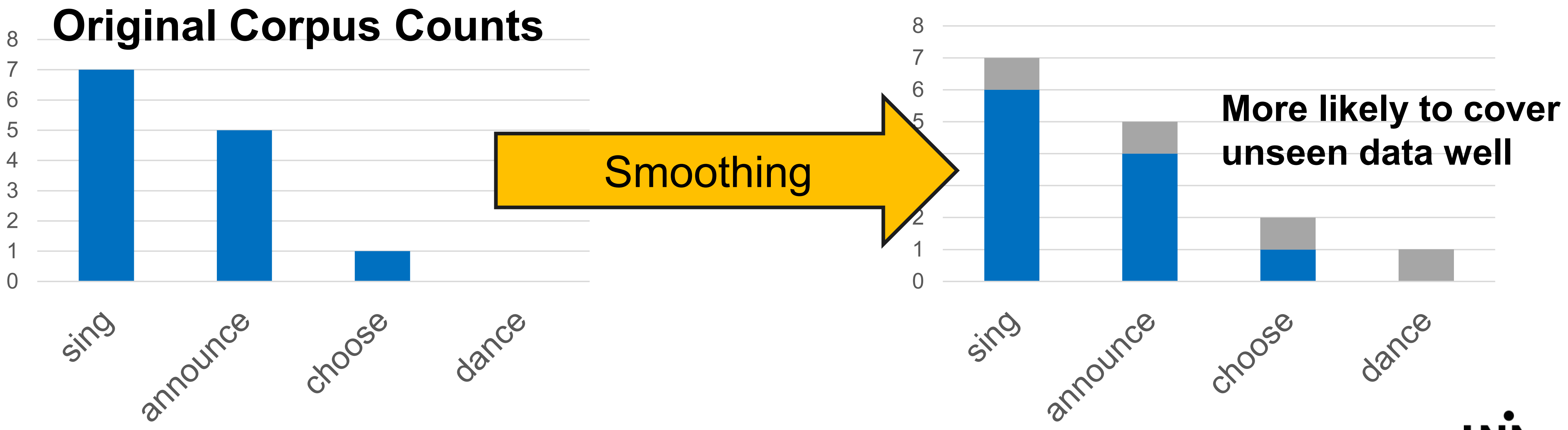
Trigram language model: $P(W) \approx \prod_i P(w_i|w_{i-2}w_{i-1})$

N-Gram Language Modeling: Smoothing

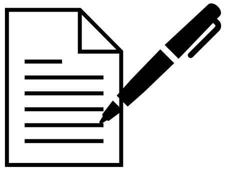
P(w | participant country to)   four-grams

$$P_{Add-1}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + V}$$

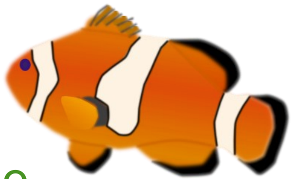
Add-1 Smoothing
(Laplace Smoothing with α = *1*)



In-Class Activity 3.1



Bigram Language Model: $P(W) = P(w_1,w_2,w_3,w_4,w_5\ldots w_n) \approx \prod P(w_i|w_{i-1})$



Fish respire from their gills.

Test set = real-world text

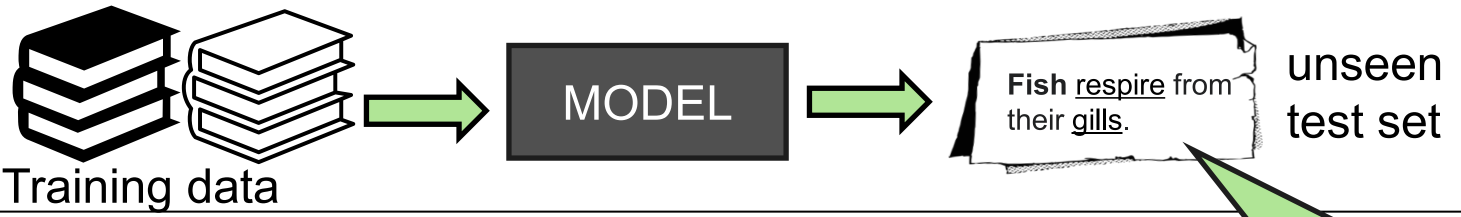
Which probabilities do the two language models assign to the real-world sentence?

Which language model captures the „real world" in a better way?

Language Model A	
P(Fish|START)	0.2
P(respire|Fish)	0.1
P(from|respire)	0.7
P(their|from)	0.2
P(gills|their)	0.05
P(.|gills)	0.5

Language Model B	
P(Fish|START)	0.1
P(respire|Fish)	0.1
P(from|respire)	0.2
P(their|from)	0.4
P(gills|their)	0.3
P(.|gills)	0.2

Perplexity



A better language model will assign a higher probability!

The best language model is one that best predicts an unseen test set
→ gives the highest P(sentence)

→ **Perplexity PP** is the inverse probability of the test set W, normalized by the number of words in the test set N → per-word metric

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} = \sqrt[N]{P(w_1 w_2 \ldots w_N)^{-1}} = \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}}$$

Cheat Sheet

$$x^{-n} = \frac{1}{x^n}$$

$$x^{\frac{1}{n}} = \sqrt[n]{x}$$

Chain rule: $$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_2 \ldots w_{i-1})}}$$

for bigrams: $$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1})}}$$

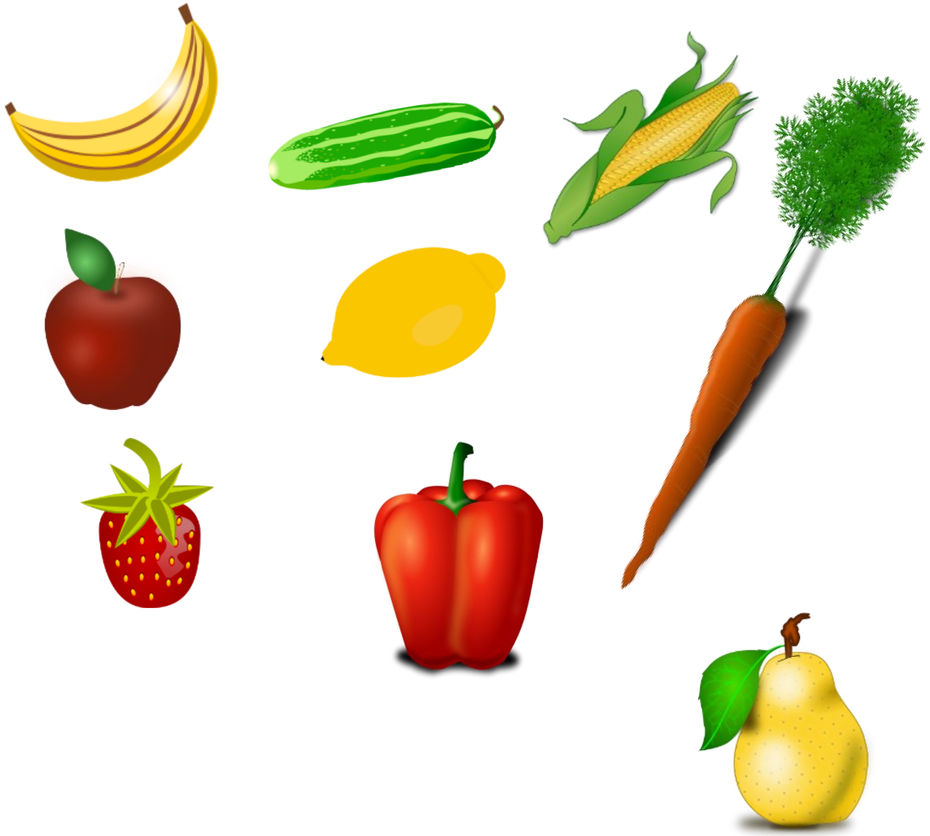
Minimizing perplexity is the same as maximizing probability!

Machine Learning Basics

In-Class Activity 3.2

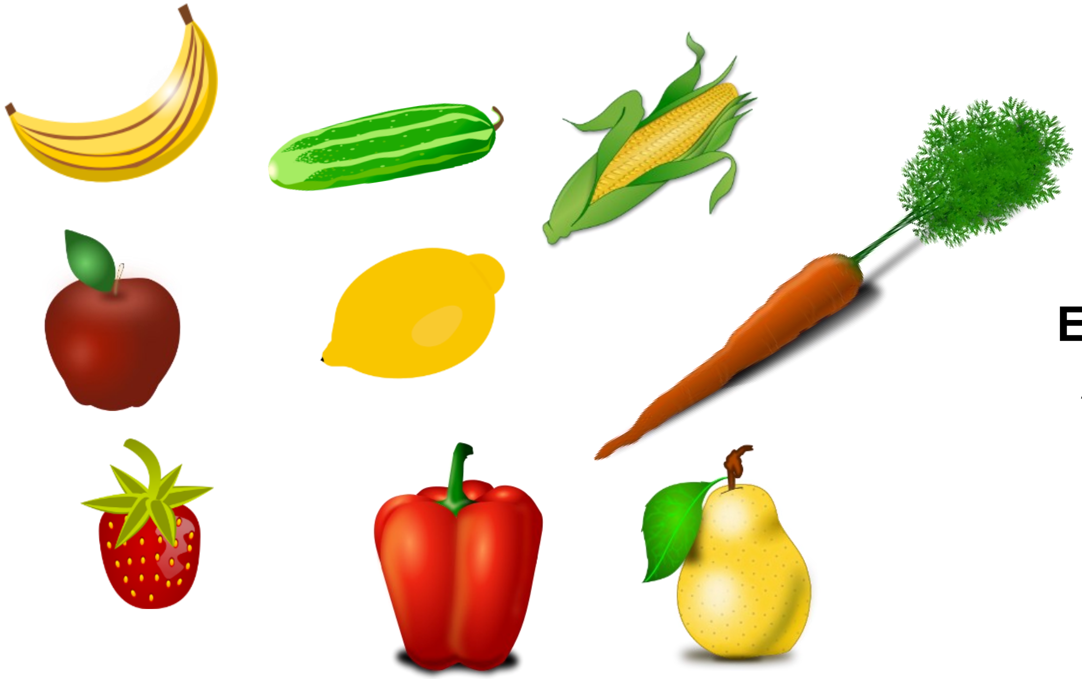
Camera

Which fruit/vegetable?



Which attributes/features can help us to distinguish these types of fruits and vegetables?

Pattern Recognition



Examples / Training instances:
Attributes/feature and values +
gold class (correct class)

Machine Learning Algorithm

Attributes:

object to be classified

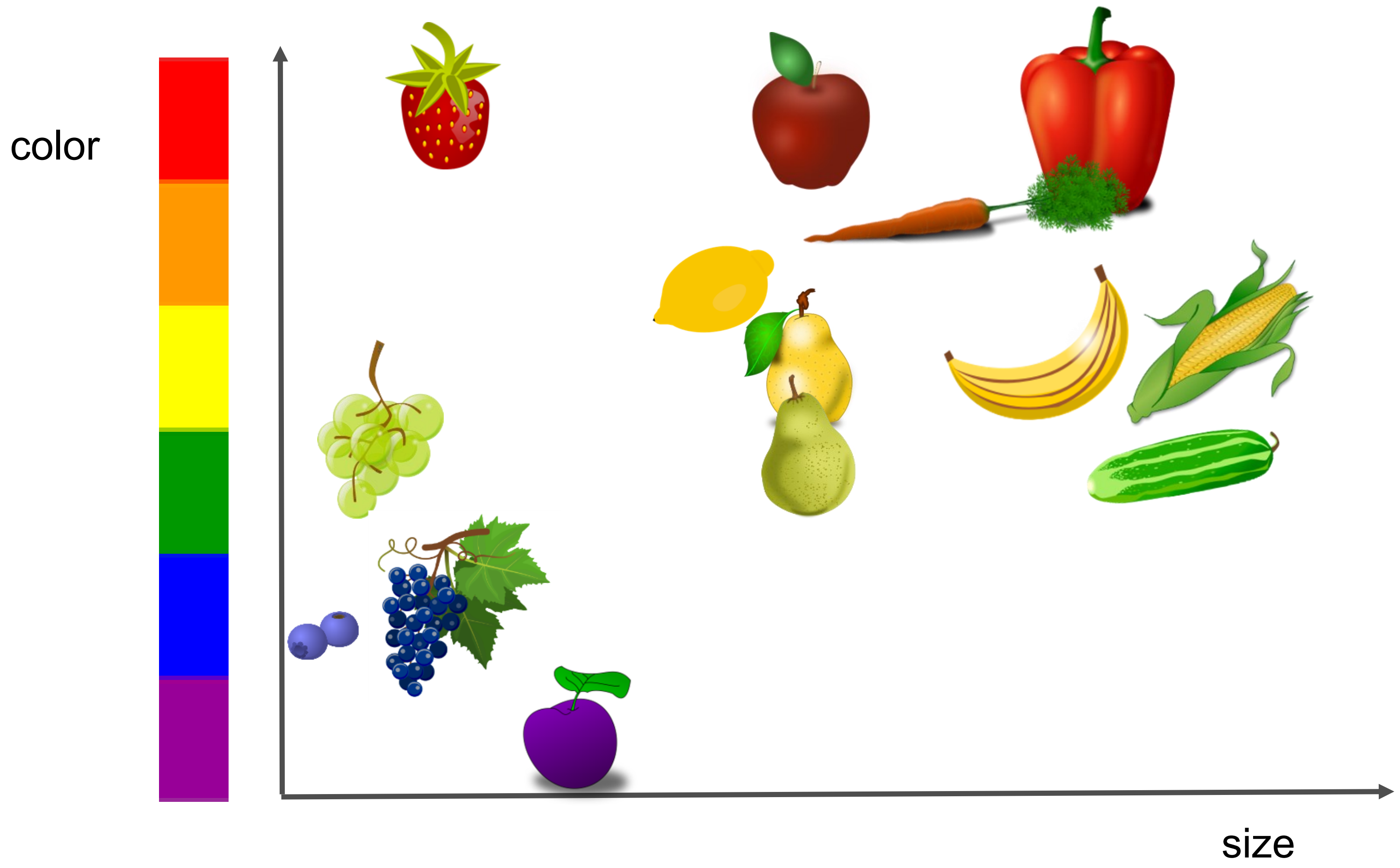
Classifier

Class label
(+ confidence)

LEMON (80%)
BANANA (20%)

Vectors / Embeddings

Represent instances in a vector space of attribute values

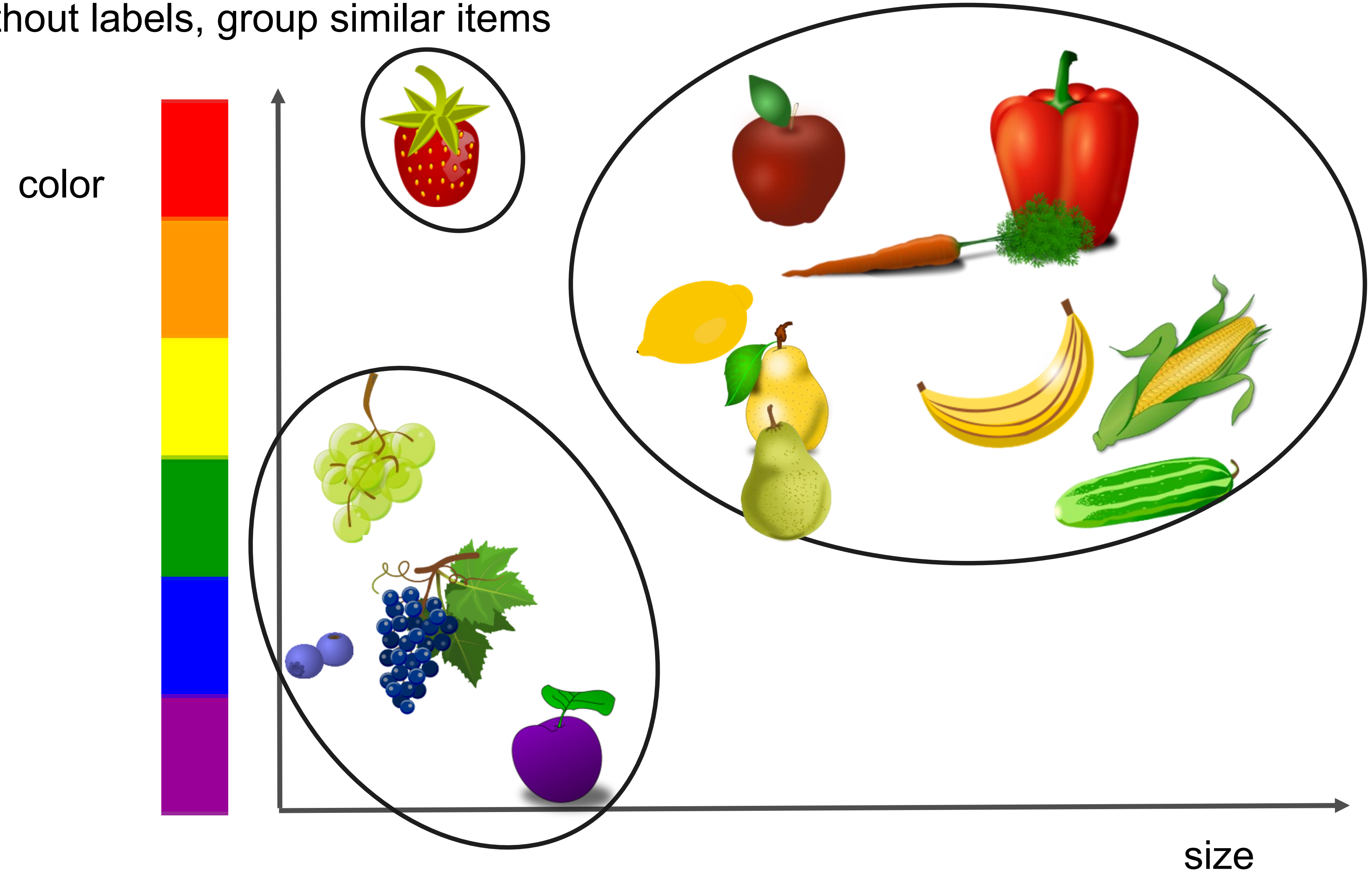


$$pepper = \begin{pmatrix} red \\ medium \end{pmatrix}$$

$$blueberry = \begin{pmatrix} blue \\ tiny \end{pmatrix}$$

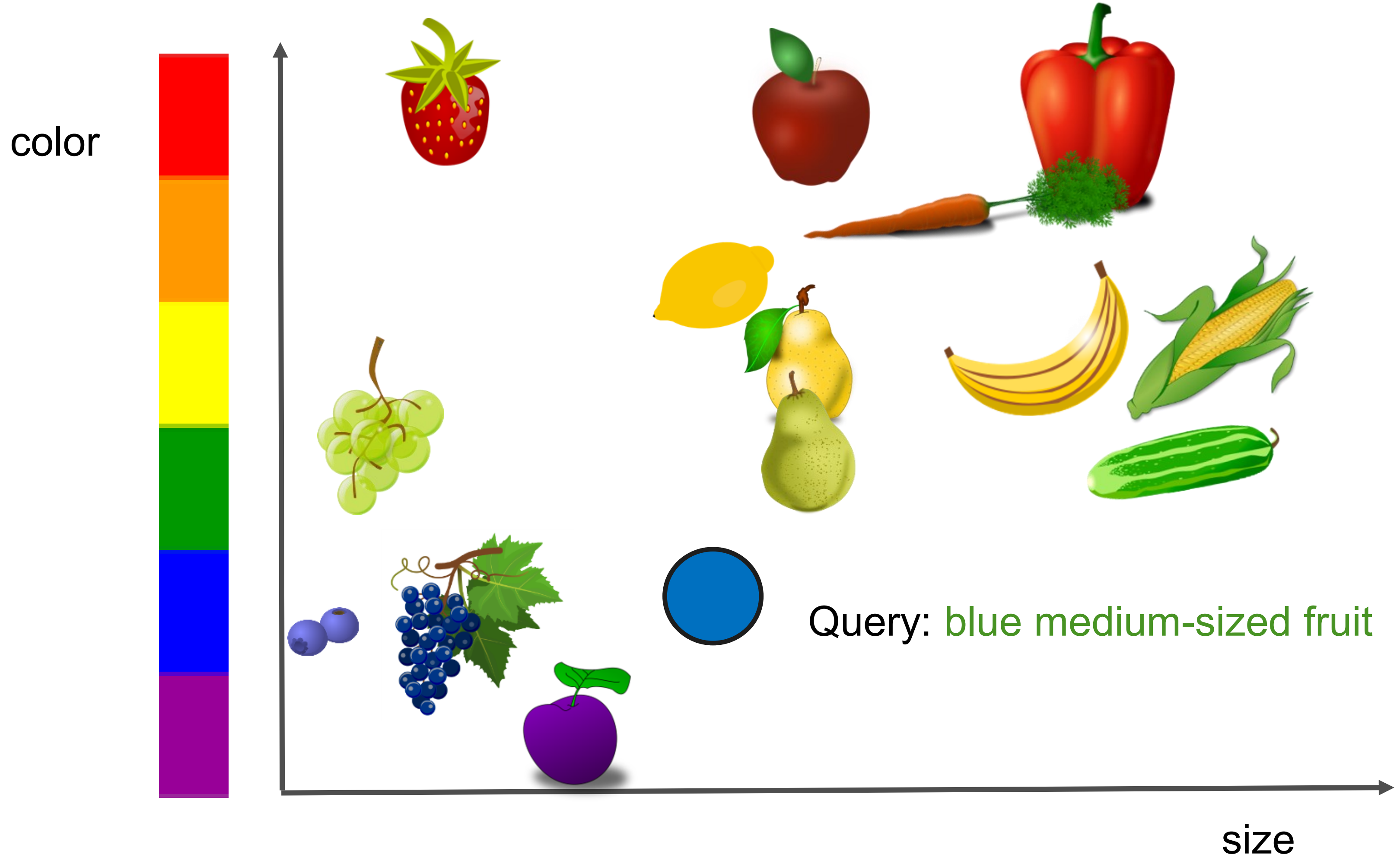
Clustering

Represent instances in a vector space of attribute values
Without labels, group similar items



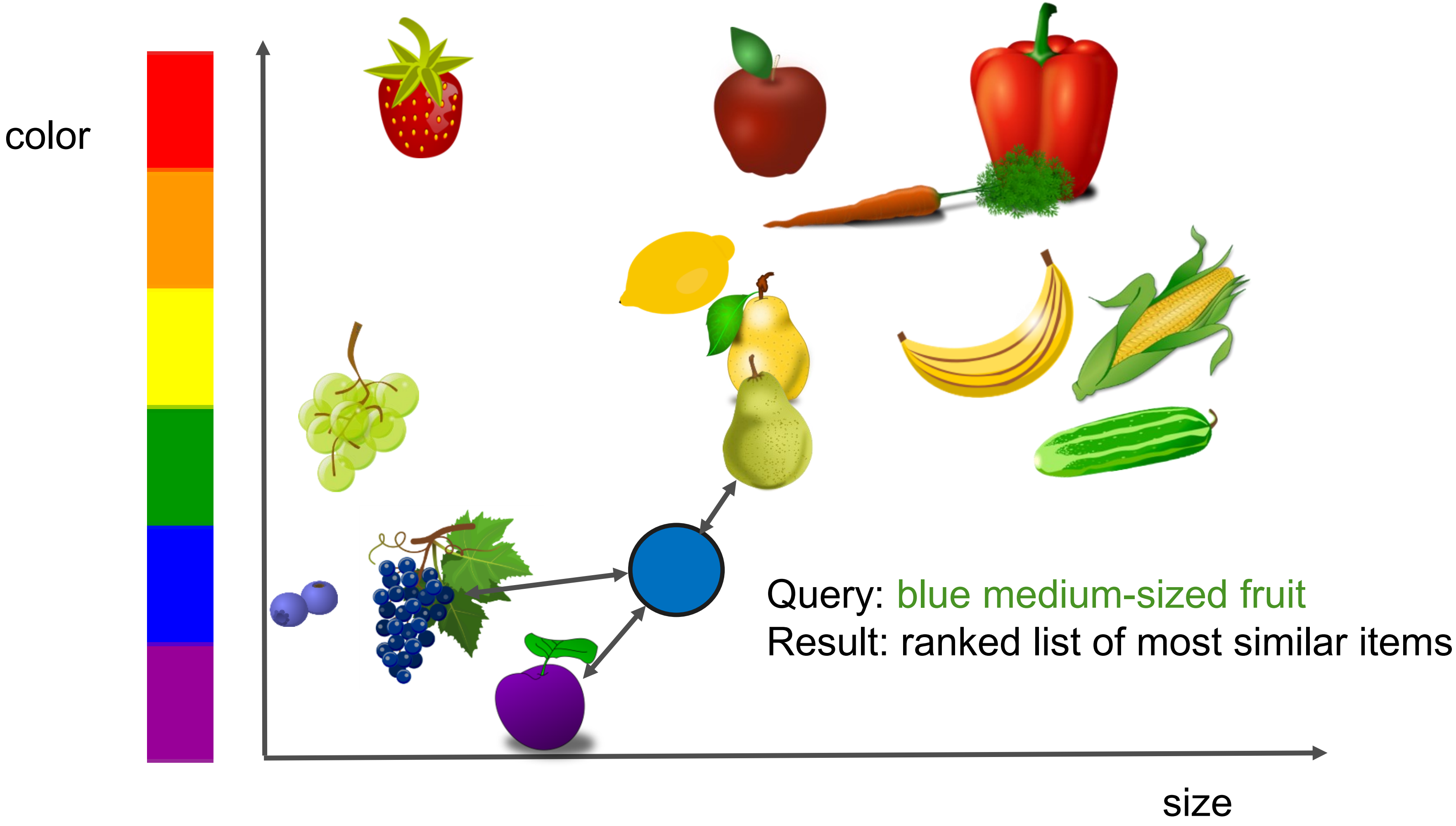
Search

Find the most similar item(s)



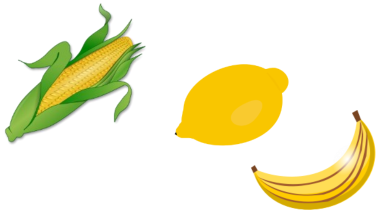
Search

Find the most similar item(s)



Types of Attributes

Discrete values



{"This fruit is long and yellow.", BANANA}
{"This fruit is yellow and sour.", LEMON}
{"This vegetable is long and yellow.", CORN}

	BANANA	LEMON	CORN
this	1	1	1
fruit	1	1	0
vegetable	0	0	1
is	1	1	1
long	1	0	0
yellow	1	1	1
and	1	1	1
sour	0	1	0

Word presence, one-hot encoding

word order has been discarded

Enumerating all words in train_data: possibly huge vocabulary – inefficient

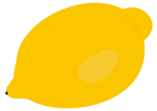
Strategies to reduce vocabulary size for discrete features:

- Choose N most frequent words (usually N~several 10k)
- Choose top N words according to tf.idf
- Use n-grams (+POS)

Types of Attributes

Numeric/real-valued attributes

{"This fruit is a little sour.", ORANGE}
{"This fruit is sour, it is extremely sour.", LEMON}



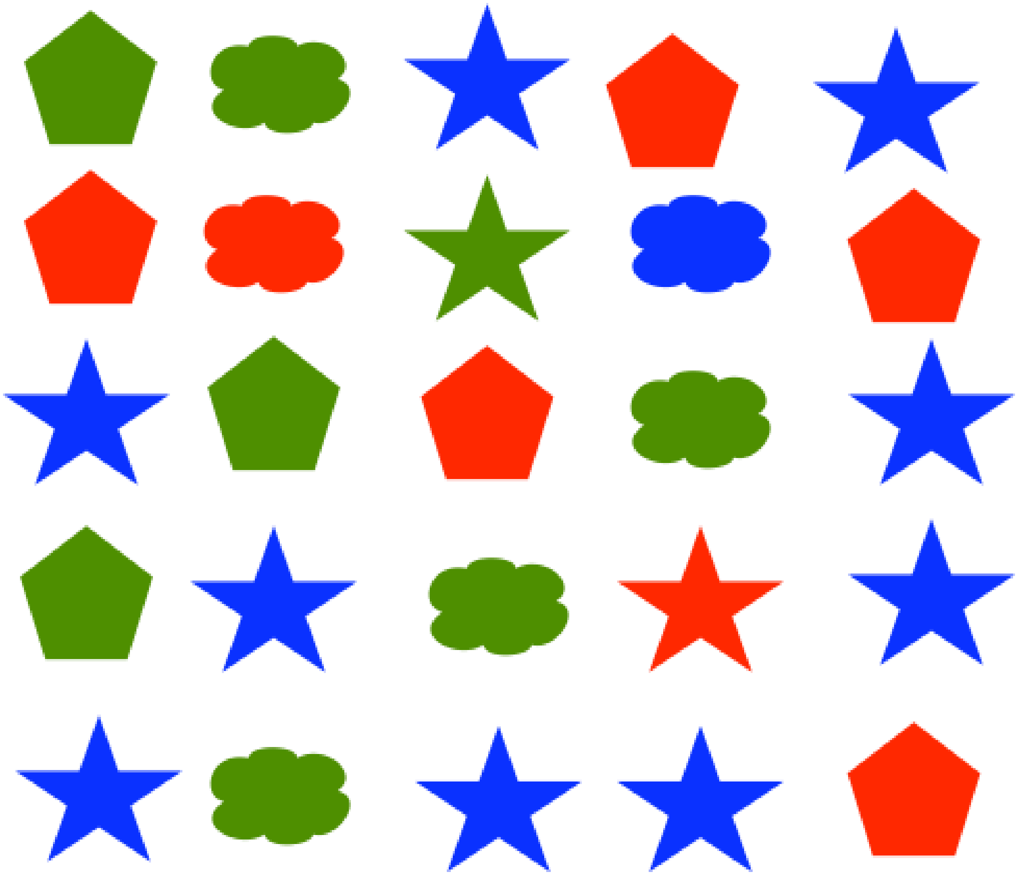
	ORANGE	LEMON
this	1	1
fruit	1	1
is	1	2
it	1	2
sour	1	2
...	…	…

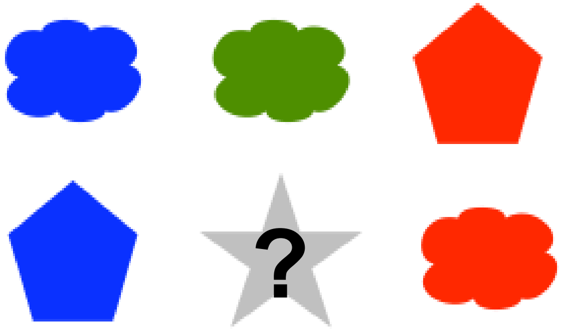
Word counts

	BANANA	LEMON	CORN
this	1	1	1
fruit	1	1	0
vegetable	0	0	1
is	1	1	1
long	1	0	0
yellow	1	1	1
and	1	1	1
sour	0	1	0

Word presence, one-hot encoding

Sparse Data Problem





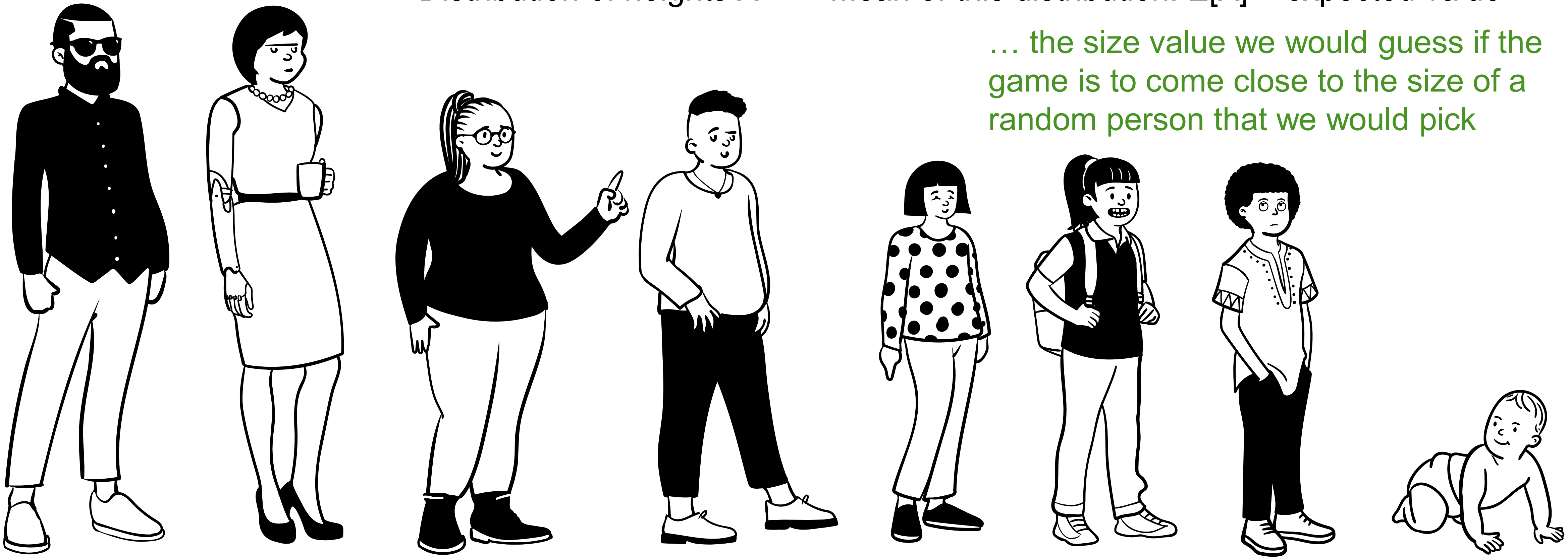
"This vegetable is purple and tastes like nothing."

Background: Expectation Values

Distribution of heights X

Mean of this distribution: E[X] = expected value

… the size value we would guess if the game is to come close to the size of a random person that we would pick



X = {1.90m, 1.90m, 1.75m, 1.75m, 1.50m, 1.50m, 1.50m, 0.70m}

$$E[X] = (1.90 + 1.90 + 1.75 + 1.75 + 1.50 + 1.50 + 1.50 + 0.70) / 8$$
$$= 2/8*1.90 + 2/8*1.75 + 3/8*1.50 + 1/8*0.70$$

Evaluation Setup and Accuracy

$Accuracy_{train}(C)$ = % of training instances correctly classified by classifier C

$Accuracy_{D}(C)$ = % of correctly classified instances in real (test) distribution

development / validation / test set

train_data	development / validation set	test set

No overlap allowed!

During development / tuning: <u>do not look at test set results</u>!

Common splits: 80/10/10, 70/15/15, ...

In-Class Activity 3.3

Connect matching boxes.

$P(w_5|w_1,w_2,w_3,w_4)$

inverse probability of a test set

$$\begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

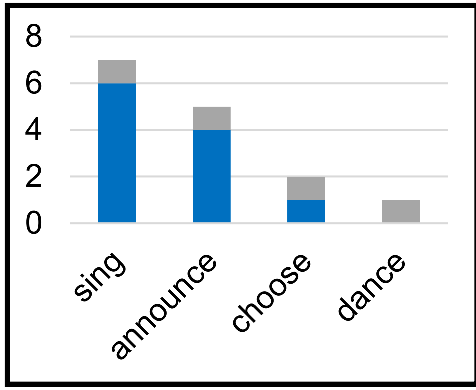
$$\sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}}$$

feature vector

to account for unseen word sequences

$P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$

$$pepper = \begin{pmatrix} red \\ medium \end{pmatrix}$$



5-gram language model probability

for checking language model quality

word sequence probability

one-hot encoding

perplexity

smoothing