CBS COPENHAGEN BUSINESS SCHOOL
HANDELSHØJSKOLEN

session-01:

Fundamentals of Machine Learning and Natural Language Processing

Sippo Rossi
Assistant Professor
Hanken School of Economics
Email: sippo.rossi@hanken.fi

Raghava Mukkamala
Associate Professor & Director,
Centre for Business Data Science
Copenhagen Business School, Denmark
Email: rrm.digi@cbs.dk, Centre: https://cbsbda.github.io/

PhD Course on Natural Language Processing and Large Language Models for Research Data Exploration and Analysis
Aalto University, Finland

EQUIS ACCREDITED AACSB ACCREDITED ASSOCIATION AMBA ACCREDITED CEMS PIM

Outline

- Fundamentals of Machine Learning
  - Supervised, Unsupervised, Reinforcement Learning
- Precision Measures and Benchmarks
- Basic Text Processing Tokenization
- Word Normalization and Stemming
- Parts of Speech Tagging
- Word Frequencies

Fundamentals of Machine Learning

CBS

Terminology

Artificial intelligence: Can mean pretty much anything where a machine does something that seemingly needs human intelligence.

Machine learning: Algorithms that learn or find patterns from data and generalize (make predictions) on new data.

Deep learning: a subset of machine learning models that are based on artificial neural networks.

Generative AI: Deep learning models that have been trained on massive datasets and can generate statistically probable outputs

Types of machine learning

ML methods can be categorized broadly based on:

How they are trained:

- unsupervised learning
- supervised learning
- reinforcement learning
- reinforcement learning with human feedback

How they work:

- instance-based
- model-based learning

Supervised learning

The training algorithm is shown training data that includes labels

E.g., training a model to classify ratings as positive or negative

Examples of algorithms:

- Linear regression
- K-nearest neighbors
- Random forest

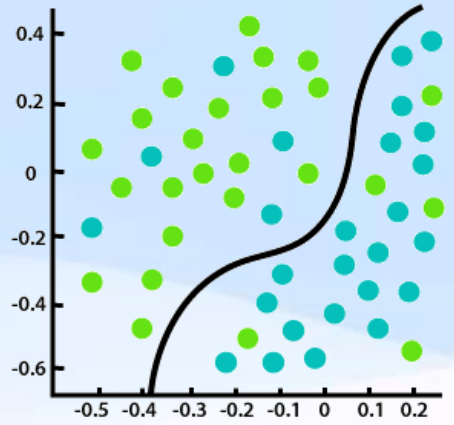
Stanford Dogs dataset
(120 breeds)

Supervised Learning: classification & regression

Differ in the types of labels (dependent variables)

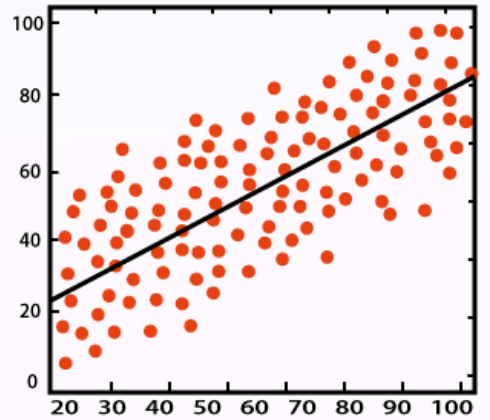
Classification:

- Discrete/categorical labels
- Text labeling, sentiment prediction



Regression:

- Continuous labels
- Stock prices, temperature, sales volume



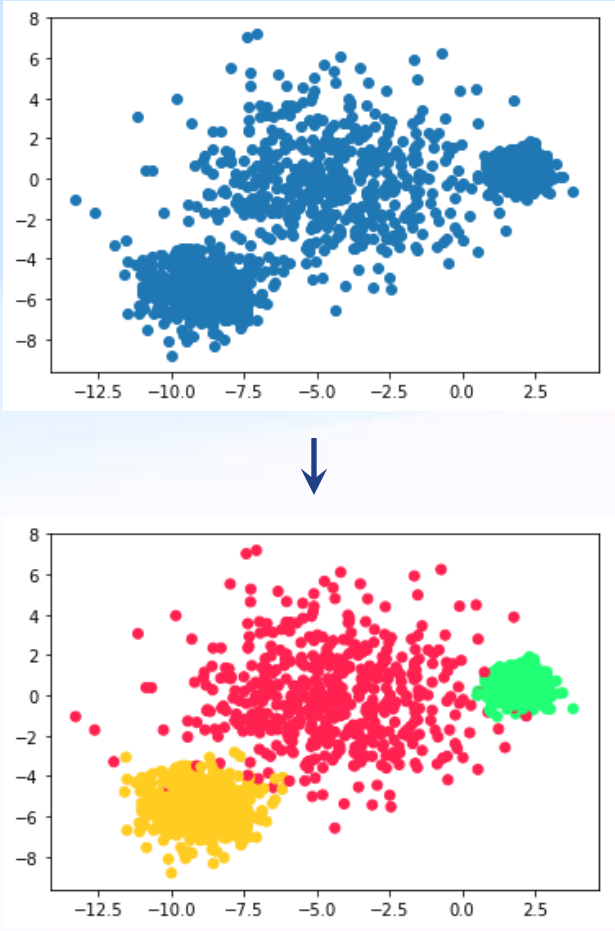
Unsupervised learning

Labels are not provided, and the algorithm learns how to group the data without being explicitly told what the groups are

E.g., topic modelling, grouping a webstore's customers into segments

Example algorithms:

- Clustering: k-means, hierarchical
- Dimensionality reduction: principal components analysis (PCA), t-SNE



Reinforcement learning

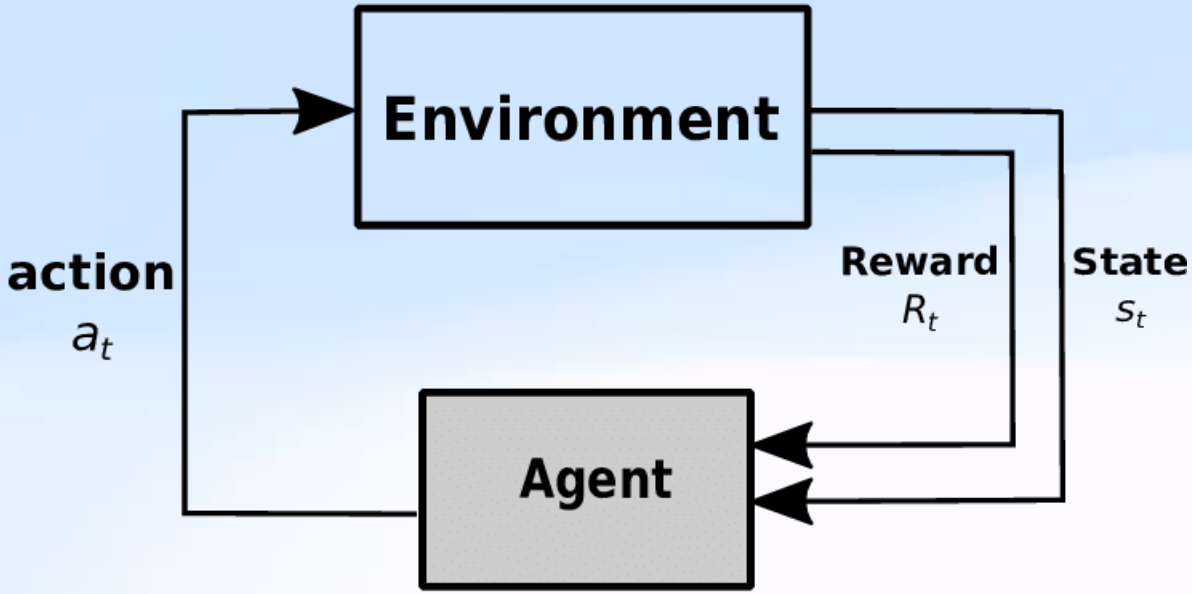


Image from A Machine Learning Approach for Power Allocation in HetNets Considering QoS, Amiri et al. 2018

Reinforcement learning

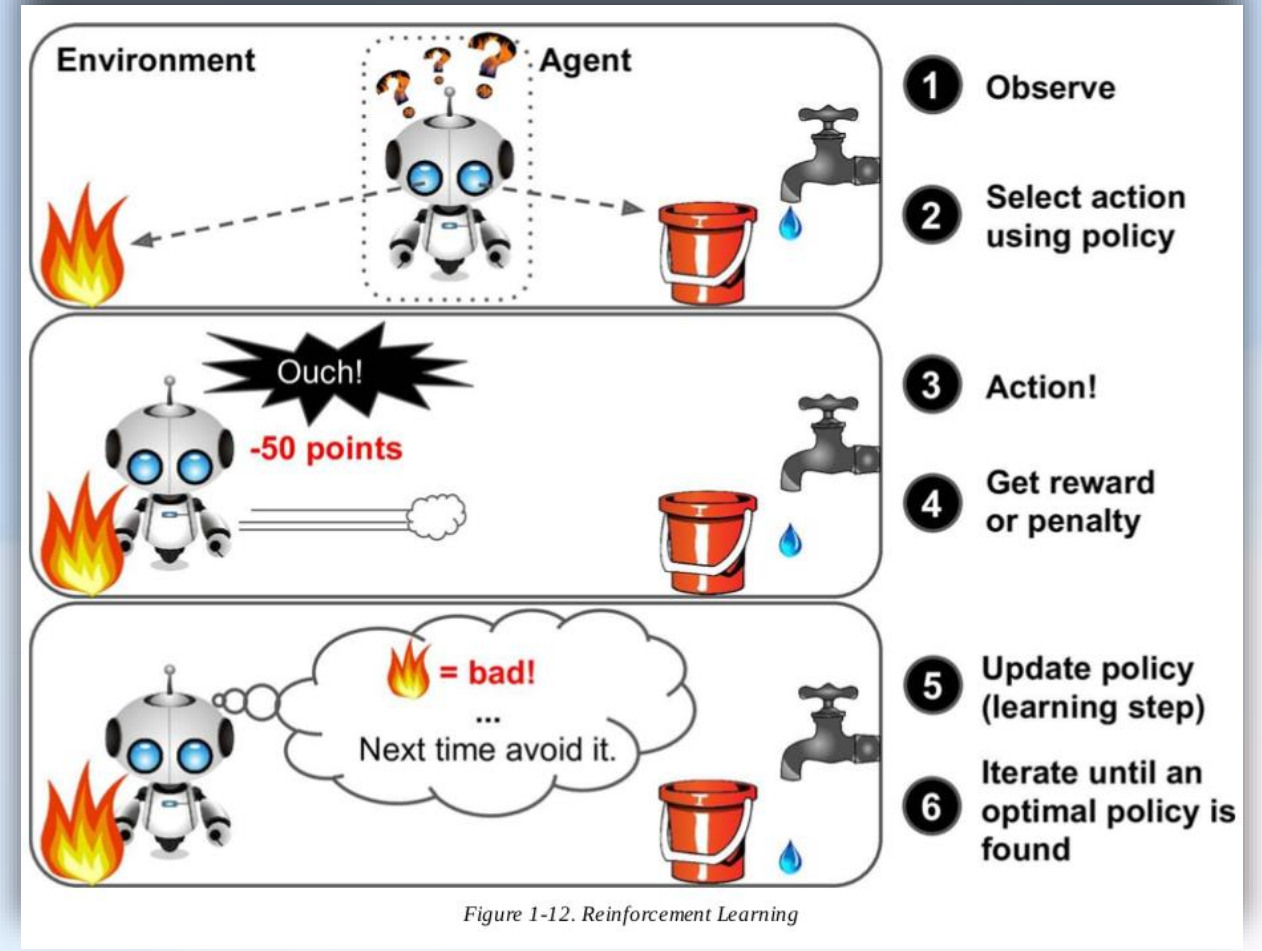


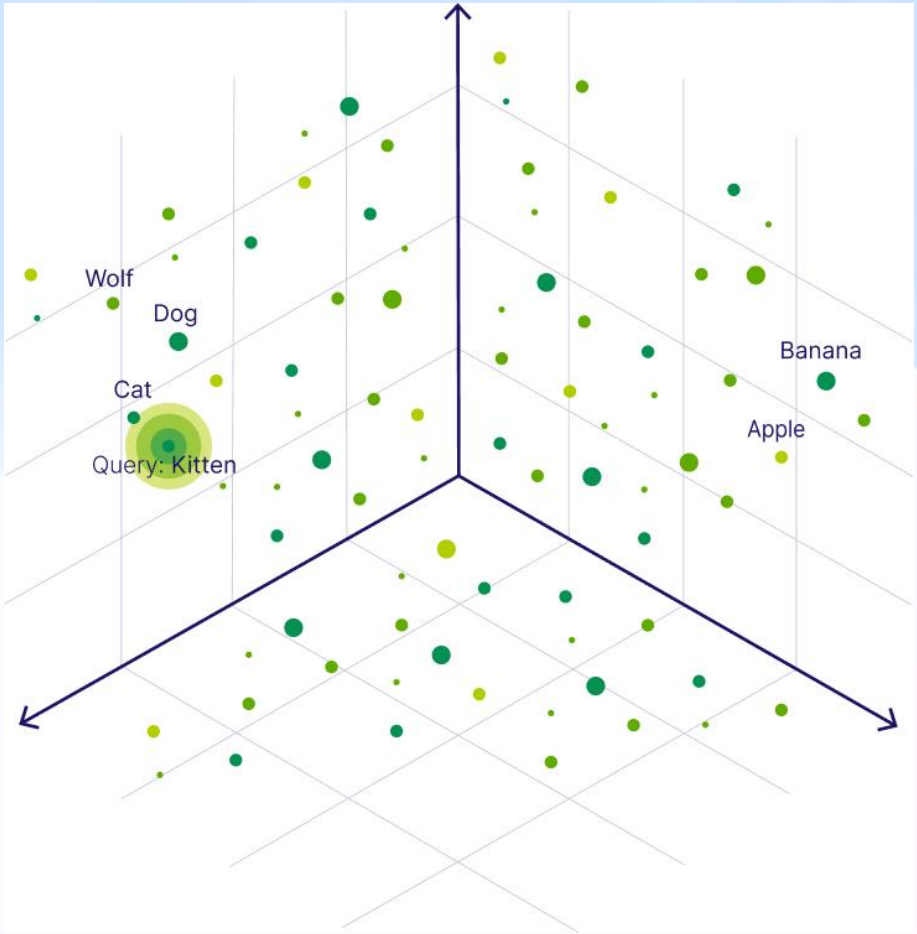
Figure 1-12. Reinforcement Learning

Source: Aurélien Géron "Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems"

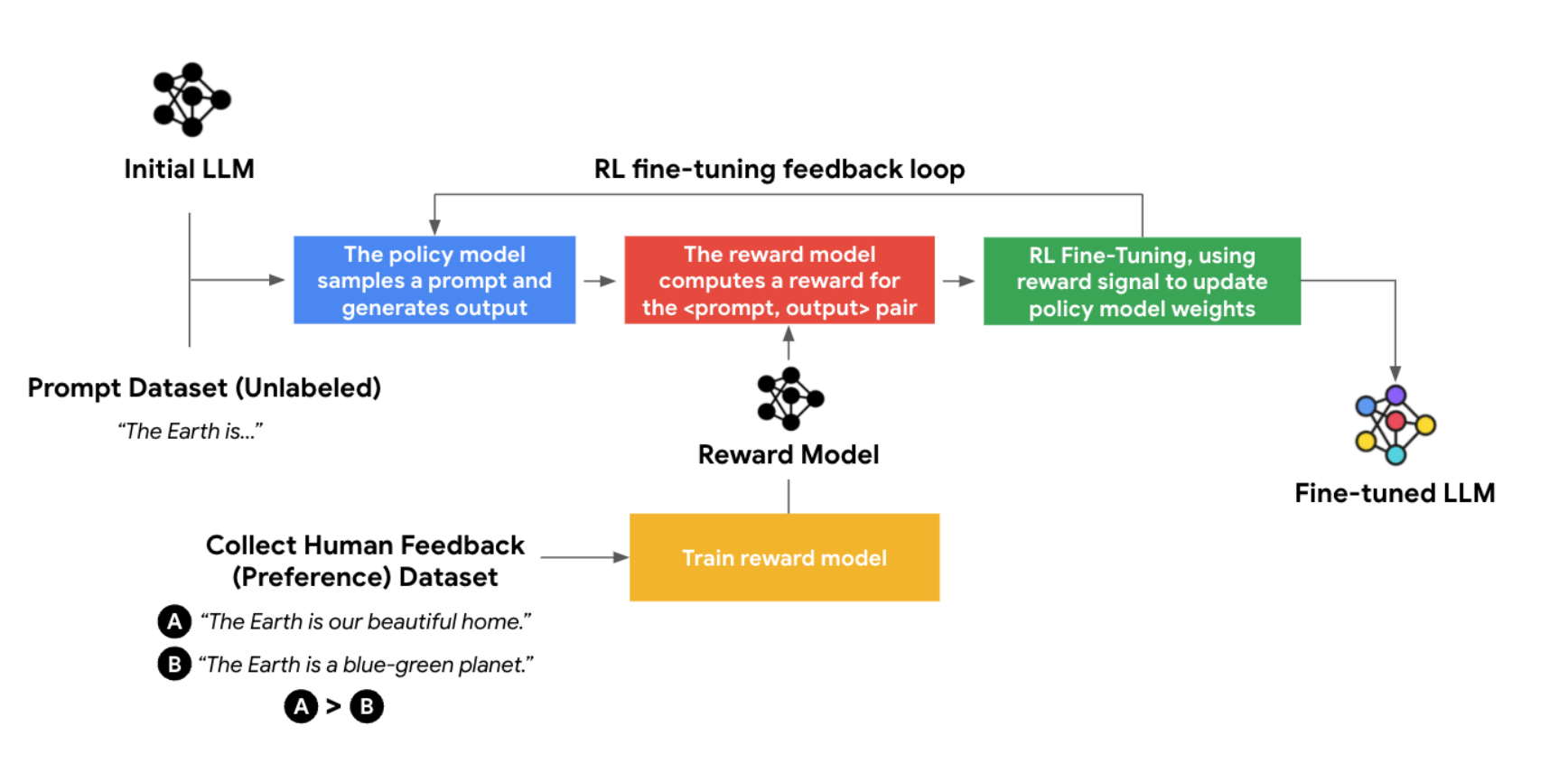
Large language models (LLM)

Are large deep learning-based models trained on massive amounts of text data and which learn the statistical relationships between texts.

Can be used e.g., for text generation, summarization and classification among other things.



Deep reinforcement learning in LLMs



Source: https://cloud.google.com/blog/products/ai-machine-learning/rlhf-on-google-cloud

How machine learning* works in practice

1. Collect data and preprocess it
2. Split a dataset into training and test data**
3. Choose an algorithm and set its hyperparameters***
4. Train a model with the training data
5. Evaluate how well the model works with the test data
6. Repeat steps 2-4 until you are satisfied with the results

* This is an example of supervised machine learning

** Why we do this is explained later

*** Hyperparameters control the learning process

How using LLMs works in practice

Application programming interfaces (APIs).

APIs are mechanisms that make it possible for several programs to communicate with each other.

In very simple terms, an API allows you to write a program (or simply a script), that interacts with for example GPT-4 (the model behind ChatGPT). This is done in code, rather than by interacting via e.g., a web app.

```
from openai import OpenAI
client = OpenAI()

completion = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[
    {"role": "system", "content": "You are a poetic assistant, skilled in explaining complex programming
    {"role": "user", "content": "Compose a poem that explains the concept of recursion in programming."}
  ]
)
```

Training Machine Learning Models

Parameters

There are two types of parameters:

1) Parameters
2) Hyperparameters

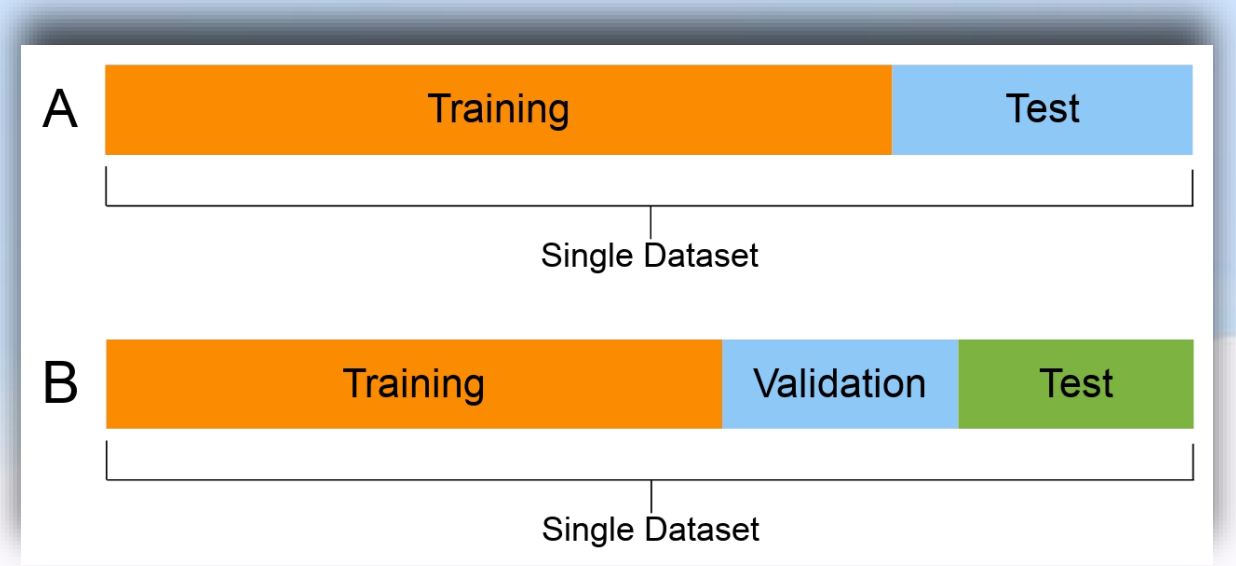
The model parameters are adjusted automatically as you fit (train) a ML model

Hyperparameters are parameters that control how a ML model learns and need to be adjusted by the user

Splitting data

The dataset is ideally divided into 3 subsets:

- Training set, to train the model
- Validation set, to tune the hyperparameters
- Test set, to confirm the results



Sometimes only two are used:

- Training set
- Test set

Never train and test a model with the same data

Image from https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets#/media/File:ML_dataset_training_validation_test_sets.png

Precision Measures and Benchmarks

Metrics to Measure Performance of ML models

- How do we know whether the algorithm did a good job or not?

Classification metrics

- Confusion matrix, precision, recall
- F1 score and accuracy

Regression metrics (not covered in this lecture)

- Mean squared error
- Root mean squared error

Confusion Matrix

		Predicted	
		Negative *(class 0)*	**Positive** *(class 1)*
Actual	**Negative** *(class 0)*	True negative TN	False positive FP
	Positive *(class 1)*	False negative FN	True positive TP

Confusion Matrix with fraud detection example

		Predicted	
		Negative *(Not fraud)*	**Positive** *(Fraud)*
Actual	**Negative** *(Not fraud)*	9900	0
	Positive *(Fraud)*	100	0

In this example, the model has predicted everything to be negative.

If measuring only for accuracy, the model would still be 99% accurate...

Notes on the previous example

One metric and especially accuracy alone is bad

If the classes are imbalanced, high accuracy can be achieved with just predicting everything to belong to the largest class

In the example accuracy was seemingly good although the model was bad and didn’t detect cases of fraud at all

Even with a balanced dataset it is always a good idea to measure and report the performance in multiple ways

Different metrics

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{FP} + \text{FN} + \text{TP}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

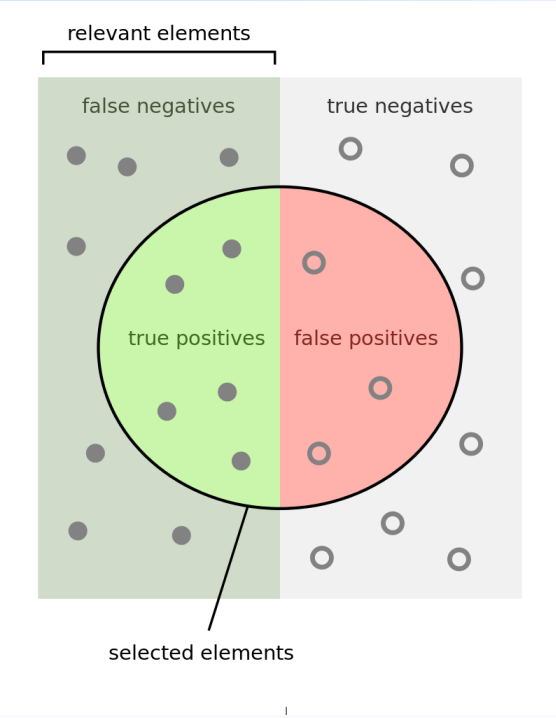
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

		Predicted	
		Negative *(class 0)*	**Positive** *(class 1)*
Actual	**Negative** *(class 0)*	True negative TN	False positive FP
	Positive *(class 1)*	False negative FN	True positive TP

Recall

Precision



How many selected items are relevant?

Precision =

How many relevant items are selected?

Recall =

Benchmarks

LLMs are typically assessed via benchmarks

Benchmarks consist of a sample set of data, questions and tasks that test the LLMs in areas such as math, translation, Q&A

Commonly a model will be tested with multiple benchmarks. There are many websites containing leaderboards with LLMs and how they perform with different benchmarks

One example of a leaderboard: https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard#/

Natural Language Processing

CBS

Natural language processing (NLP)

NLP as a topic is old and vast, although it became a topic of general interest only recently due to generative AI

One definition of NLP:
“a subfield of computer science and AI that use ML to enable computers to understand and communicate with human language”

We will start by going through some fundamental aspects and methods used in NLP, before going to the more modern examples

Text is high dimensional and sparse data

Imagine that you have data about 60 Facebook posts with features:

- #likes, #comments, #shares, #comment-replies
- For the text associated with each post, use a bag of words approach!
  - Combine texts from all posts and prepare a dictionary ($\Longrightarrow$ 300 unique words)
  - words in the columns indicate how many times each word has occurred in the post

$$\mathbf{Z} = \begin{bmatrix} 6 & 2 & 1 & 8 \\ 12 & 21 & 0 & 2 \\ 26 & 19 & 0 & 2 \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ 2 & 9 & 7 & 2 \\ 7 & 13 & 81 & 2 \\ 11 & 9 & 38 & 2 \end{bmatrix}_{60\times 4}$$

Facebook Post Attributes

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 & 0 & \cdots & \cdots & 2 & 0 & 3 \\ 0 & 0 & 2 & \cdots & \cdots & 1 & 0 & 0 \\ 1 & 0 & 0 & \cdots & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 2 & \cdots & \cdots & 1 & 0 & 0 \\ 3 & 1 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 3 & 0 & \cdots & \cdots & 0 & 0 & 1 \end{bmatrix}_{60\times 300}$$

Text Features of Facebook Posts

	Terms				
	Camera	Digital	Memory	Print	...
Document 1	3	2	0	1	
Document 2	0	4	0	3	
...	...	...	...	...	

Approaches to dealing with text: bag of words & string of words

```
#Example 1: Bag of words and string of words
"the theology is the theory of the religion"

# Bag_of_words approach:
bag_of_words = {'the', 'theology', 'is', 'theory','of','
   religion'}

# String_of_words approach:
string_of_words = {('the',0), ('theology',1), ('is',2), ('the'
   ,3), ('theory',4) ,('of',5),('the',6),('religion',7)}
```

- Why is it so important?
  - Shakespeare plays contain 885 000 words, but the count of unique words is 31 000

Text normalization

Every NLP task needs to do text normalization

Normalizing text includes:

1. Segmenting/tokenizing words in running text
2. Normalizing word formats
3. Segmenting sentences in running text

Basic Text Processing Tokenization

Many of the slides have been taken and adapted from: Speech and Language Processing by Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

CBS

Types and tokens

Type: the class of all tokens containing the same character sequence.

Token: an instance of a sequence of characters in a document

”they lay back on the San Francisco grass and looked at the stars and their"

How many?

- 15 tokens (or 14)
- 13 types (or 12) (or 11?)

How many words?

N = number of tokens

V = vocabulary = set of types

$|V|$ is the size of the vocabulary

Church and Gale (1990): $|V| > O(N^{½})$

	Tokens = N	**Types = |V|**
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million

Based on slides by Dan Jurafsky

Issues in tokenization

- Finland's capital → Finland Finlands Finland's ?
- what're, I'm, isn't → What are, I am, is not
- Hewlett-Packard → Hewlett Packard ?
- state-of-the-art → state of the art ?
- Lowercase → lower-case lowercase lower case ?
- San Francisco → one token or two?
- m.p.h., PhD. → ??

Based on slides by Dan Jurafsky

Word Normalization and Stemming

Many of the slides have been taken and adapted from: Speech and Language Processing by Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

CBS

Normalizing terms

Need to “normalize” terms

- Information Retrieval: indexed text & query terms must have same form.
  - We want to match ***U.S.A.*** and ***USA***

We implicitly define equivalence classes of terms

- e.g., deleting periods in a term

Alternative: asymmetric expansion:

- Enter: ***window*** Search: ***window, windows***
- Enter: ***windows*** Search: ***Windows, windows, window***
- Enter: ***Windows*** Search: ***Windows***

Potentially more powerful, but less efficient

Based on slides by Dan Jurafsky

Case folding

Applications like information retrieval: reduce all letters to lower case

- Since users tend to use lower case
- Possible exception: upper case in mid-sentence?
  - e.g., ***General Motors***
  - ***Fed*** vs. ***fed***
  - ***SAIL*** vs. ***sail***

For sentiment analysis, machine translation, information extraction

- Case is helpful (***US*** versus ***us*** is important)

Based on slides by Dan Jurafsky

Lemmatization

Lemmatization: have to find correct dictionary headword form

Reduce inflections or variant forms to base form

- *am, are, is => be*
- *car, cars, car's, cars' => car*
- *the boy's cars are different colors => the boy car be different color*

Machine translation

- Spanish quiero ('I want'), quieres ('you want') same lemma as querer 'want'

Based on slides by Dan Jurafsky

Stemming

Reduce terms to their stems in information retrieval

Stemming is crude chopping of affixes

- language dependent
- e.g., ***automate(s), automatic, automation*** all reduced to ***automat***.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Based on slides by Dan Jurafsky

Sentence Segmentation

Many of the slides have been taken and adapted from: Speech and Language Processing by Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

CBS

Sentence Segmentation

!, ? are relatively unambiguous

Period "." is quite ambiguous

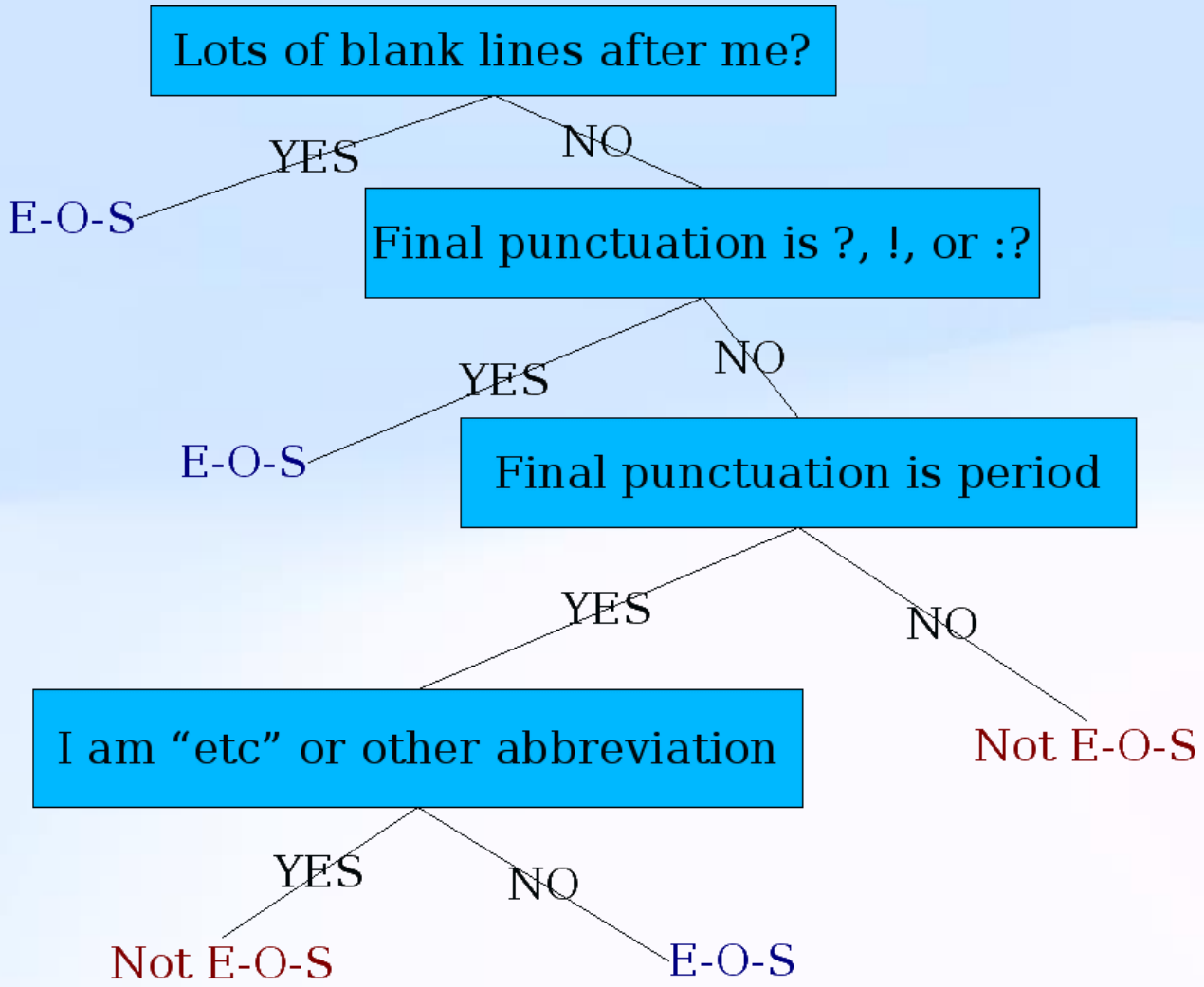
- Sentence boundary
- Abbreviations like Inc. or Dr.
- Numbers like .02% or 4.3

Build a binary classifier

- Looks at a "."
- Decides EndOfSentence/NotEndOfSentence
- Classifiers: hand-written rules, regular expressions, or machine-learning

Based on slides by Dan Jurafsky

Determining if a word is end-of-sentence



Based on slides by Dan Jurafsky

Parts of Speech Tagging

Many of the slides have been taken and adapted from: Speech and Language Processing by Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

CBS

Open vs. closed classes

Open vs. closed classes

Closed:

- determiners: ***a, an, the***
- pronouns: ***she, he, I***
- prepositions: ***on, under, over, near, by, …***
- Why “closed”?

Open:

- Nouns, Verbs, Adjectives, Adverbs
- e.g: iPhone, fax, tweeting

Based on slides by Dan Jurafsky

Parts of Speech (POS)

Open class (lexical) words

- Nouns (NN)
  - Proper (NNP): *IBM*, *Italy*
  - Common: *cat / cats*, *snow*
- Verbs (VB)
  - Main: *see*, *registered*
  - Modals: *can*, *had*
- Adjectives (JJ) *old older oldest*
- Adverbs (RB) *Slowly, unfortunately*
- Numbers (CD): *122,312*, *one*

… more

Closed class (functional)

- Determiners (DT) *the some*
- Conjunctions (CC) *and or but*
- Pronouns (PRP) *he its*
- Prepositions (IN) *to with*
- Particles (RP) *off up*
- Interjections (UH) *Ow Eh*

… more

POS Tags: http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Source: https://web.stanford.edu/~jurafsky/NLPCourseraSlides.html https://web.stanford.edu/~jurafsky/slp3/9.pdf

Part of Speech (POS) Tagging

Words often have more than one POS: *back*

- The *back* door = Adjective
- On my *back* = Noun
- Win the voters *back* = Adverb
- Promised to *back* the bill = Verb

The POS tagging problem is to determine the POS tag for a particular instance of a word.

Thank you!

Raghava Mukkamala

rrm.digi@cbs.dk

www.cbs.dk/staff/rrmdigi

raghavamukkamala.github.io/

https://cbsbda.github.io/

Sippo Rossi

sippo.rossi@hanken.fi

www.hanken.fi/en/person/sippo-rossi