Text Mining and Language Analytics

Lecture 2

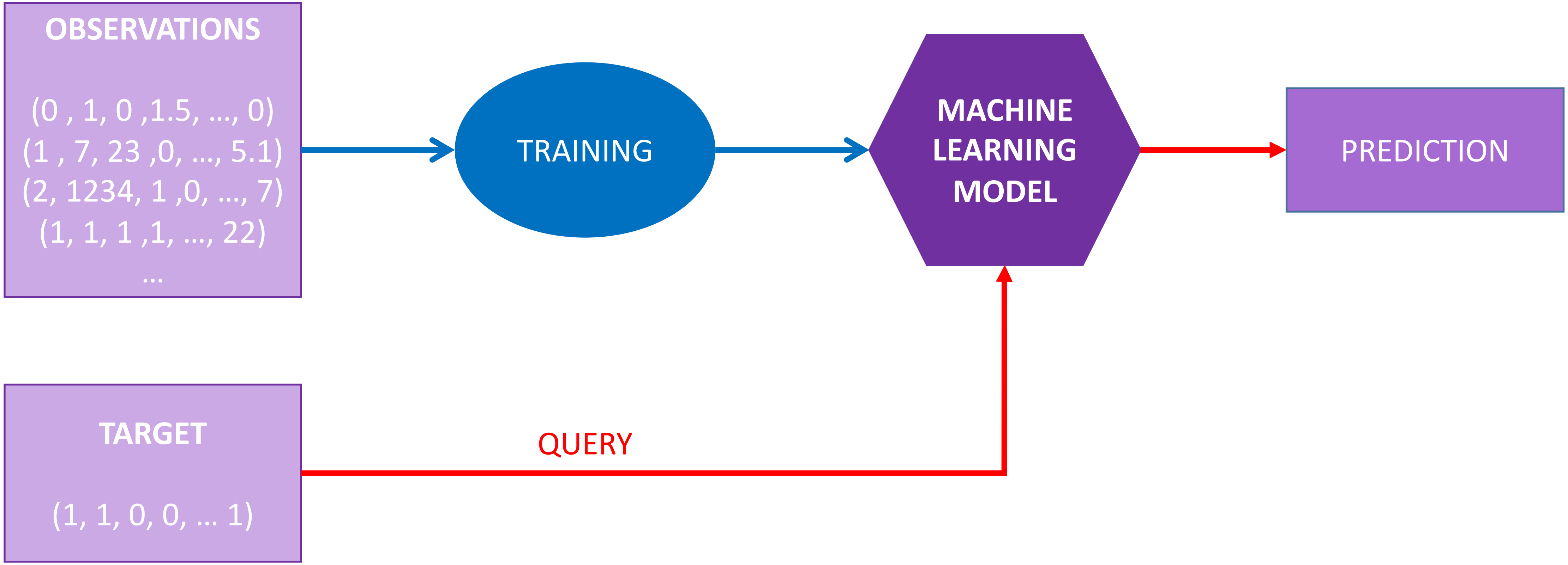
One-hot encoding & Term frequency-based representation

Dr Stamos Katsigiannis

2023-24

Durham University

Machine learning paradigm



Durham University

Text encoding for machine learning

- Consider the sentence "The red table is broken"
- Let's apply tokenisation, stop words removal and lemmatisation
  - Result → [red, table, break]
- Typical machine learning algorithms require numerical vectors
- How can we convert [red, table, break] to a numerical vector?

Durham University

One-Hot representation (I)

- Consider the following two sentences
  - The table is red
  - The blue table is broken

- Tokenising these two sentences yields 6 unique tokens
  - {The, table, is, red, blue, broken}
  - i.e. A vocabulary of size 6

- Each token (word) can be represented by a 6-dimensional vector

One-Hot representation (II)

Word	The	table	is	red	blue	broken	Vector
The	1	0	0	0	0	0	(1, 0, 0, 0, 0, 0)
table	0	1	0	0	0	0	(0, 1, 0, 0, 0, 0)
is	0	0	1	0	0	0	(0, 0, 1, 0, 0, 0)
red	0	0	0	1	0	0	(0, 0, 0, 1, 0, 0)
blue	0	0	0	0	1	0	(0, 0, 0, 0, 1, 0)
broken	0	0	0	0	0	1	(0, 0, 0, 0, 0, 1)

The one-hot representation for a sentence, phrase or document is the logical OR between its constituent words

The table is red → (1, 1, 1, 1, 0, 0)

The blue table is broken → (1, 1, 1, 0, 1, 1)

Durham University

One-Hot representation (III)

The	$\rightarrow$	(1, 0, 0, 0, 0, 0)
table	$\rightarrow$	(0, 1, 0, 0, 0, 0)
is	$\rightarrow$	(0, 0, 1, 0, 0, 0)
red	$\rightarrow$	(0, 0, 0, 1, 0, 0)

OR (1, 1, 1, 1, 0, 0)

The	$\rightarrow$	(1, 0, 0, 0, 0, 0)
blue	$\rightarrow$	(0, 0, 0, 0, 1, 0)
table	$\rightarrow$	(0, 1, 0, 0, 0, 0)
is	$\rightarrow$	(0, 0, 1, 0, 0, 0)
broken	$\rightarrow$	(0, 0, 0, 0, 0, 1)

OR (1, 1, 1, 0, 1, 1)

One-Hot representation (IV)

Text 1
The old table is red but it is broken. We will use the blue table until it is fixed.

Text 2
The old blue table was fixed but we will use the red table which is broken.

Stop words
the, is, but, it, we, will, the, until, which

Tokenisation
Stop words removal
One-Hot encoding

Word	Text 1	Text 2
old	1	1
table	1	1
red	1	1
broken	1	1
use	1	1
blue	1	1
fixed	1	1

Text 1 → (1, 1, 1, 1, 1, 1, 1)

Text 2 → (1, 1, 1, 1, 1, 1, 1)

Notice anything strange?

One-Hot representation (V)

- One-Hot representation does not take into account relationships between words, their order or their frequency of occurrence
- Also known as the Bag of Words approach

- Information about order and structure within a text is discarded
- Measures the presence of known words out of a vocabulary

Durham University

Term Frequency (TF) representation (I)

- Consider the following two sentences:
  - I saw a red table, a red car, and a red box
  - I saw a table and a car and a red box
- Tokenisation leads to a vocabulary of size 8:
  - {I, saw, a, red, table, car, and, box}
- Using one-hot encoding:
  - I saw a red table, a red car, and a red box → (1, 1, 1, 1, 1, 1, 1)
  - I saw a table and a car and a red box → (1, 1, 1, 1, 1, 1, 1)
- One-hot encoding fails to capture the difference in occurrences of the words "red" and "and"

Durham University

Term Frequency (TF) representation (II)

- TF representation → The sum of the one-hot representations of a text's constituent words
  - $tf(t, d)$ → frequency of term *t* in document *d*

- The TF representation of the two previous sentences would be:

		I	saw	a	red	table	car	and	box
I saw a red table, a red car, and a red box	→	(1,	1,	3,	3,	1,	1,	1,	1)
I saw a table and a car and a red box	→	(1,	1,	3,	1,	1,	1,	2,	1)

Durham University

What about short vs. large documents?

Term Frequency (TF) representation (III)

Variants of TF computation:

Weighting scheme	$tf(t,d)$	
Binary	0, 1	One-hot encoding
Frequency	$f(t,d)$	Raw count (typical TF representation)
Normalised frequency by the number of words in d	$f(t,d) \Big/ \sum_{t' \in d} f(t',d)$	Raw count divided by the number of words in d
Log normalisation	$\log(1+f(t,d))$	Prevents large values
Augmented frequency	$0.5 + 0.5 \cdot \frac{f(t,d)}{\max\{f(t',d) : t' \in d\}}$	Prevents bias towards longer documents by dividing with the frequency of the most occurring term in d

d: document (text)
t: unique term
$f(t,d)$: frequency of term t in document d

Document Frequency (DF)

- Rare terms are more informative than frequent terms
- Frequent terms are less informative than rare terms
  - e.g. stop words

- High weight for rare terms needed

- Document frequency $df(t, D)$ → The number of documents that contain the term t in corpus D
  - Inverse measure of term t's informativeness
  - $df(t, D) \leq N$, N number of documents in corpus D

High $df(t, D)$ for frequent terms

Durham University

Term Frequency - Inverse Document Frequency (TF-IDF) (I)

- Document frequency can be used to penalise terms that are frequent across the documents in a corpus

- Inverse document frequency of term t in corpus D

$$idf(t, D) = \log\left(\frac{N}{df(t,D)}\right)$$

- TF-IDF of term t in document d of corpus D

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$

- **High TF-IDF** requires a **high term frequency** (in a given document) and a **low document frequency** of the term across the documents in the corpus

Durham University

Term Frequency - Inverse Document Frequency (TF-IDF) (II)

- Common terms across documents are penalised

- What about stop words?

- TF-IDF can be used to rank "important" terms of each document in a collection (corpus)
  - Match queries with most relevant documents ← Search engines!
  - Extract document keywords ← Word clouds
  - Topic modelling ← Text categorisation

Durham University

Features comparison

- Consider the following sentences:
  - **Text 1:** I saw a red table, a red car, and a red box
  - **Text 2:** I bought a red table

Terms	One-hot		TF		TF normalised by no. of terms		TF log normalised		TF augmented frequency		DF	IDF	TF-IDF	
	T1	T2	T1	T2	T1	T2	T1	T2	T1	T2			T1	T2
I	1	1	1	1	0.08	0.2	0.3	0.3	0.67	1	2	0	0	0
saw	1	0	1	0	0.08	0	0.3	0	0.67	0.5	1	0.3	0.3	0
a	1	1	3	1	0.25	0.2	0.6	0.3	1	1	2	0	0	0
red	1	1	3	1	0.25	0.2	0.6	0.3	1	1	2	0	0	0
table	1	1	1	1	0.08	0.2	0.3	0.3	0.67	1	2	0	0	0
car	1	0	1	0	0.08	0	0.3	0	0.67	0.5	1	0.3	0.3	0
and	1	0	1	0	0.08	0	0.3	0	0.67	0.5	1	0.3	0.3	0
box	1	0	1	0	0.08	0	0.3	0	0.67	0.5	1	0.3	0.3	0
bought	0	0	0	1	0	0.2	0	0.3	0.5	1	1	0.3	0	0.3

*Values rounded to the 2nd decimal

Durham University

Documents as vectors (I)

- Documents (texts) are now represented by real-valued vectors of weights $\in \mathbf{R}^{|V|}$
  - $|V|$-dimensional vector space

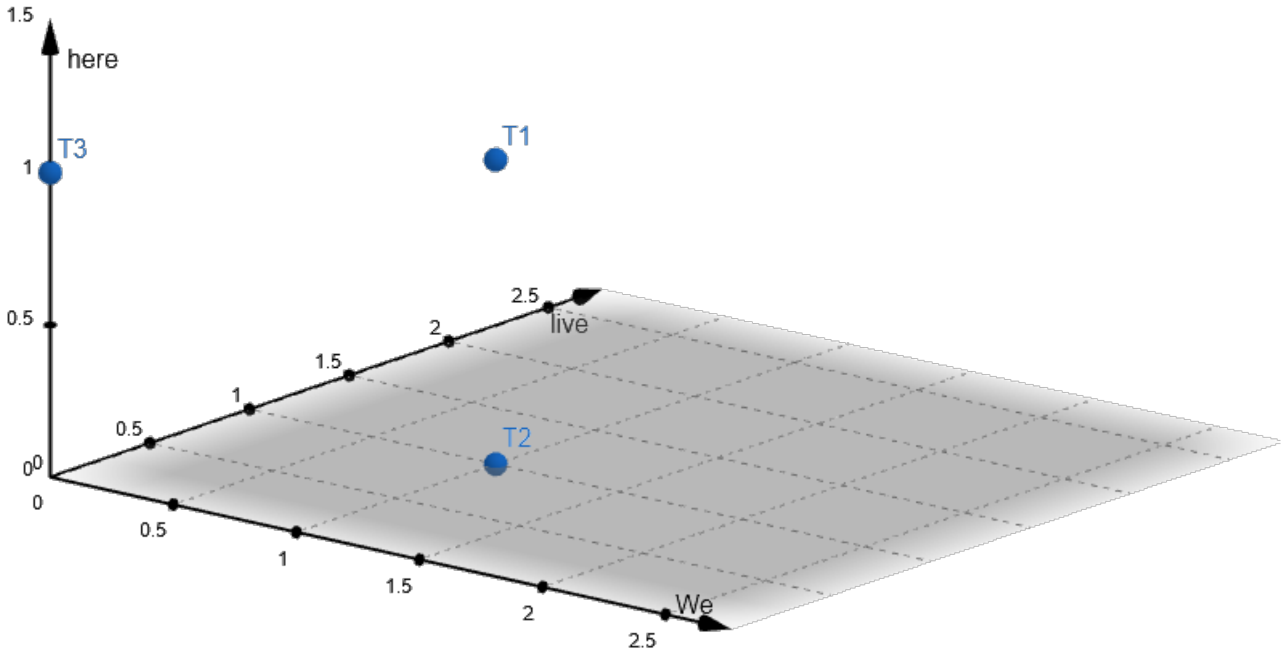
- Terms are axes of the space
- Documents (texts) are points in this space
  - Very high dimensional representation
  - e.g. Millions of dimensions for a web search engine

- Very sparse vectors → **Most entries are zero**

Durham University

Documents as vectors (II)

- One-hot encoding of the following sentences
  - **T1**: We live here → (1, 1, 1)
  - **T2:** We live → (1, 1, 0)
  - **T3:** here → (0, 0, 1)

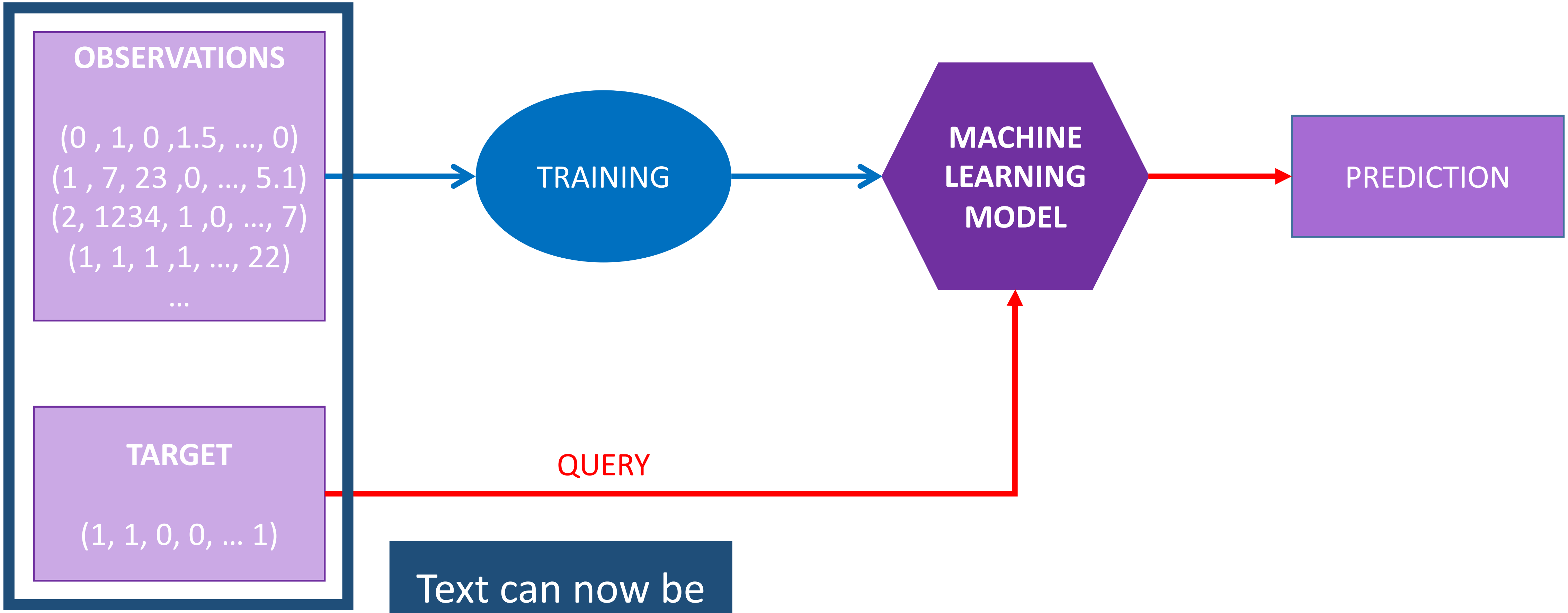


- Vocabulary: {We, live, here}

- 3-dimensional vector space

Durham University

Machine learning paradigm (again)



Text can now be used as an input!

Questions?

Durham University