WORKING WITH TEXT DATA - PART I

Practical Lectures

by Chris Emmery (MSc)

TODAY'S LECTURE

- Representing text as vectors.
- Binary vectors for Decision Tree classification.
- Using Vector Spaces and weightings.
- Document classification using k-NN.

HOW IS THIS DIFFERENT THAN BEFORE?

- Numbers are numbers. Their scales and distributions might be different; the information leaves little to interpretation.
- Language is complex:
  - Representing language is complex.
  - Mathematically interpreting language is complex.
  - Inferring knowledge from language is complex.
  - Understanding language is complex.

NOISY LANGUAGE

Just netflixed pixels, best time ever lol - 1/5

LANGUAGE AS A STRING

```
title,director,year,score,budget,gross,plot
"Dunkirk","Christopher Nolan",2017,8.4,100000000,183836652,"Allied soldiers
"Interstellar","Christopher Nolan",2014,8.6,165000000,187991439,"A team of e
"Inception","Christopher Nolan",2010,8.8,160000000,292568851,"A thief, who s
"The Prestige","Christopher Nolan",2006,8.5,40000000,53082743,"After a tragi
"Memento","Christopher Nolan",2000,8.5,9000000,25530884,"A man juggles searc
```

TEXT TO VECTORS

CONVERTING TO NUMBERS

$$d = \text{the cat sat on the mat} \rightarrow \vec{d} = \langle ? \rangle$$

WORDS AS FEATURES

$$d = \text{the cat sat on the mat} \rightarrow$$

$$\begin{bmatrix} \text{cat} & \text{mat} & \text{on} & \text{sat} & \text{the} \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Bag-of-Words Representation

DOCUMENTS AS INSTANCES

$$d_0 = \text{the cat sat on the mat}$$

$$d_1 = \text{my cat sat on my cat}$$

$$\begin{bmatrix} \text{cat} & \text{mat} & \text{my} & \text{on} & \text{sat} & \text{the} \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

DOCUMENTS * TERMS

$$V = [\text{cat} \quad \text{mat} \quad \text{my} \quad \text{on} \quad \text{sat} \quad \text{the}]$$

$$X = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

DOCUMENT SIMILARITY

Wikipedia articles:

data	language	learning	mining	text	vision	y
1	0	1	0	0	1	CV
1	1	1	0	1	0	NLP
1	0	1	1	1	0	TM

- CV = Computer vision
- NLP = Natural Language Processing
- TM = Text Mining

DOCUMENT SIMILARTY - JACCARD COEFFICIENT

$$d_0 = \langle 1, 0, 1, 0, 0, 1 \rangle$$
$$d_1 = \langle 1, 1, 1, 0, 1, 0 \rangle$$
$$d_2 = \langle 1, 0, 1, 1, 1, 0 \rangle$$

$$J(d_0, d_1) = 2/5 = 0.4$$
$$J(d_0, d_2) = 2/5 = 0.4$$
$$J(d_1, d_2) = 3/5 = 0.6$$

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

*words in A **and** B (intersection) /*
*words in A **or** B (union)*

DECISION TREES (ID3)

CLASSIFICATION RULES

data	language	learning	mining	text	vision	y
1	0	1	0	0	1	CV
1	1	1	0	1	0	NLP
1	0	1	1	1	0	TM

```
if 'vision' in d:
    label = 'CV'
else:
  if 'language' in d:
    label = 'NLP'
  else:
    label = 'TM'
```

INFERRING RULES (DECISIONS) BY INFORMATION GAIN

INFERRING RULES (DECISIONS) BY INFORMATION GAIN

INFERRING RULES (DECISIONS) BY INFORMATION GAIN

		spam	ham	total
free	0	1	3	4
	1	4	1	5

$$E(\text{free}, \text{Y}) = \sum_{c \in Y} P(c) E(c)$$

$$E(\text{free}{=}0, Y) = -\left(\frac{1}{4} \cdot \log_2 \frac{1}{4}\right) - \left(\frac{3}{4} \cdot \log_2 \frac{3}{4}\right)$$

$$E(\text{free}{=}1, Y) = -\left(\frac{4}{5} \cdot \log_2 \frac{4}{5}\right) \cdot -\left(\frac{1}{5} \cdot \log_2 \frac{1}{5}\right)$$

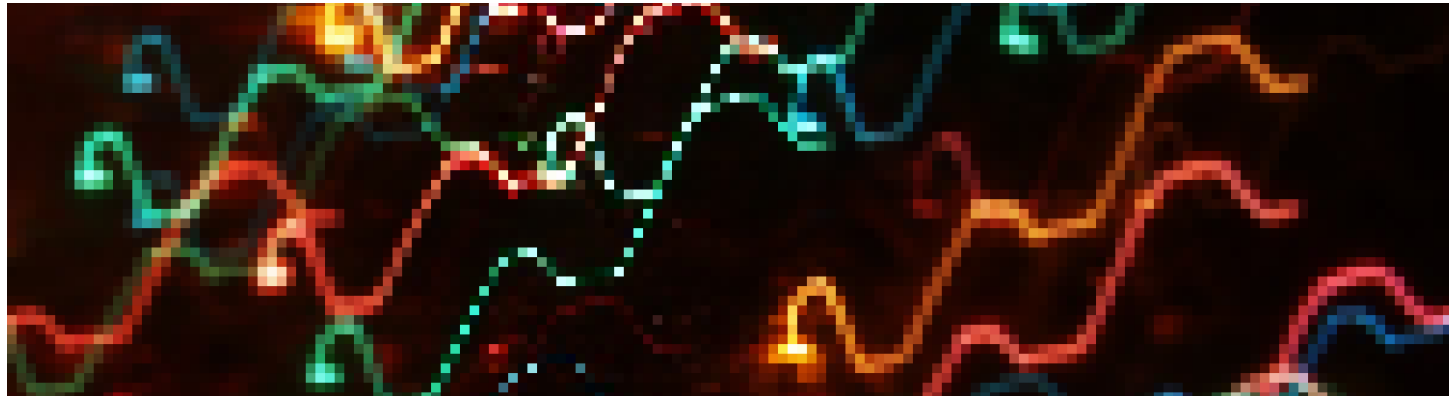
$$E(\text{free}, \text{Y}) = \frac{4}{9} \cdot 0.811 + \frac{5}{9} \cdot 0.722 = 0.762$$

ID3 ALGORITHM

- Feature (word) with highest Information Gain will be split on first.
- Instances divided over both sides calculations will be repeated on leftovers (recursion).
- If all leftover instances belong to one class, make decision.
- Create more and more rules until some stopping criterion.

More info: here, here and here.

WORDS IN VECTOR SPACES



BINARY VS. FREQUENCY

- (+) Binary is a very compact representation (in terms of memory).
- (+) Algorithms like Decision Trees have a very straight-forward and compact structure.
- (-) Binary says very little about the weight of each word (feature).
- (-) We can't use more advanced algorithms that work with Vector Spaces.

TERM FREQUENCIES - SOME NOTATION

Let $\mathbf{D} = \{d_1, d_2, \ldots, d_N\}$ be a set of documents, and $\mathbf{T} = \{t_1, t_2, \ldots, t_M\}$ (previously V) a set of index terms for $\mathbf{D}$.

Each document $d_i \in \mathbf{D}$ can be represented as a frequency vector:

$$\vec{d}_i = \langle \mathrm{tf}(t_1, d_i), \ldots, \mathrm{tf}(t_M, d_i) \rangle$$

where $\mathrm{tf}(t, d)$ denotes the frequency of term $t_j \in \mathbf{T}$ for document d_i.

Thus, $\sum_{j=1}^{J} d_j$ would be the word length of some document d.

TERM FREQUENCIES

$$d_0 = \text{the cat sat on the mat}$$

$$d_1 = \text{my cat sat on my cat}$$

$$T = \begin{bmatrix} \text{cat} & \text{mat} & \text{my} & \text{on} & \text{sat} & \text{the} \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 2 & 0 & 2 & 1 & 1 & 0 \end{bmatrix}$$

TERM FREQUENCIES?

```
d0 = 'natural-language-processing.wiki'
d1 = 'information-retrieval.wiki'
d2 = 'artificial-intelligence.wiki'
d3 = 'machine-learning.wiki'
d4 = 'text-mining.wiki'
d5 = 'computer-vision.wiki'
```

$$t = [\,\text{learning}\,]$$

$$X_t = \begin{bmatrix} 27 \\ 2 \\ 46 \\ 134 \\ 6 \\ 10 \end{bmatrix} \quad \log(X_t) = \begin{bmatrix} 3.33 \\ 1.10 \\ 3.85 \\ 4.91 \\ 1.95 \\ 2.40 \end{bmatrix}$$

- $\text{tf} = 10$ less important than $\text{tf} = 100$, but also *10?
- Information Theory to the rescue!
- $\log(\text{tf}(t, d) + 1)$
- Notice +1 smoothing to avoid $\log(0) = -\text{inf}$

SOME REMAINING PROBLEMS

- The longer a document, the higher the probability a term will occur often, and will thus have more weight.
- Rare terms should actually be informative, especially if they occur amongst few documents.
    - If d_1 and d_2 both have cross-validation in their vectors, and all the other documents do not $\rightarrow$ strong similarty.

Latter: Document Frequency

(INVERSE) DOCUMENT FREQUENCY

$$\mathrm{idf}_t = \log_b \frac{N}{\mathrm{df}_t}$$

$$t = [\mathrm{naive}]$$

$$X_t = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 3 \\ 0 \\ 0 \end{bmatrix} \quad \mathrm{df}_t = 3 \quad \mathrm{idf}_t = \log_b \frac{6}{3} = 0.30$$

PUTTING IT TOGETHER: $\mathrm{tf} * \mathrm{idf}$ WEIGHTING

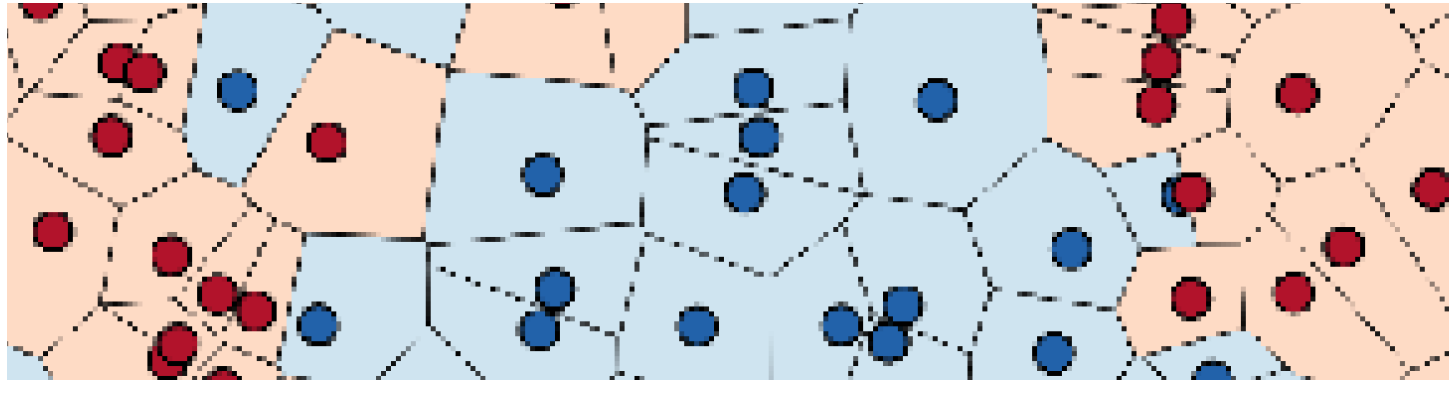
$$w_{t,d} = \log(\mathrm{tf}(t,d) + 1) \cdot \log_b \frac{N}{\mathrm{df}_t}$$

d	learning	text	language	intelligence
0	5 $\rightarrow$ 0.32	1	10	0
1	2	21 $\rightarrow$ 0.0	6	0
2	0	3	0	1 $\rightarrow$ 0.33

NORMALIZING VECTOR REPRESENTATIONS

- We fixed the global information per document $*$ term instance.
- Despite tf $*$ idf, we still don't account for the **length** of documents (i.e. the amount of words in total).
- Why is this an issue?

k-NEAREST NEIGHBOURS



EUCLIDEAN DISTANCE

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{n} (\vec{x}_i - \vec{y}_i)^2}$$

Documents with many words are far away.

ℓ_2 NORMALIZATION

$$||\vec{x}||_2 = \sqrt{\sum_i x_i^2}$$

Divide all feature values by norm.

COSINE SIMILARITY

$$\vec{a} \bullet \vec{b} = \sum_{i=1}^{n} \vec{a}_i \vec{b}_i = \vec{a}_1 \vec{b}_1 + \vec{a}_2 \vec{b}_2 + \ldots + \vec{a}_n \vec{b}_n$$

Under the ℓ_2 norm only (otherwise normalize vectors before)!

USING SIMILARITY IN k-NN

- Store the complete training matrix X_{train} in memory.
- Calculate cosine / euclidean metric between a given $\vec{x}_{\text{test}}$ and all $\vec{x}_{\text{train}} \in X_{\text{train}}$.
- Choose the k vectors from X_{train} with the highest similarity to $\vec{x}_{\text{test}}$.
- Look up the labels for these k vectors, take majority label $\rightarrow$ this is the classification.

AUGMENTATIONS TO k-NN

- Use different metrics.
- Weight labels by:
  - Frequency (majority preferred).
  - Inverse frequency (rarer preferred).
  - Distance (closer instances count heavier).

Weightings avoid ties!