# Business Analytics

## Lecture 4: Text Similarity

Ulrich Wohak[1]

[1]Department of Economics
Vienna University of Economics and Business

# Introduction

- In todays lecture we will try to answer a deceptive simple question: How similar are two documents?
- For humans this is often an easy task: ((cat, dog), (dog, animal), (dog, rocket ship))
- Are the tuple elements similar?
- *How* similar are they?
- How can we frame this question in a way that computers give us an answer?

# Introduction (ii)

- We will frame this question s.t. we can map our input text into 'document vectors'
- Then we can apply techniques from linear algebra
- Before we do this, we will need to introduce a few new concepts
- We've already encounter the simplest form of such document vectors before: the (binary) BoW representation
- Why might this be too simplistic?

# From BoW to frequencies

- We will want to augment our binary BoW representation to account for token frequency because:
    1. The frequency of a token in a document is a good indicator for its **relevance**
    2. The **relative frequency** of a token with respect to all other documents in the corpus gives an even better notion of importance
- We will augment our binary vector representations to a 'counter' vector representation
- What problem might such a representation (mechanically) induce?

# From BoW to frequencies

- To formalize our discussion a little bit, let's introduce: *term frequency (tf)*
- Generally speaking, there is strong (positive) correlation between the length of a document and the tf for a particular token
- Longer documents $\rightarrow$ higher frequencies
- Short documents $\rightarrow$ lower frequencies
- Hence, our 'count', or tf, should depend on document length!

# Example

- Consider the following counts for the token 'dog' in two documents:
  1. $tf('dog', d_1) = 3$
  2. $tf('dog', d_2) = 100$

     $\rightarrow$ Token 'dog' seems more important in $d_2$ than $d_1$
- Do we change our opinion if we also consider that:
  1. $d_1$ is an email by a veterinarian with $\text{len}(d_1) = 30$ and
  2. $d_2$ is Tolstoy's *War & Pease* with $\text{len}(d_2) = 580,000$
- With this, we can compute the 'normalised' tf
  1. $ntf('dog', d_1) = \frac{3}{30} = 0.1$
  2. $ntf('dog', d_2) = \frac{100}{580k} = 0.00017$

$\rightarrow$ These can be considered probabilities!

# Inverse Document Frequency

- So far we only considered normalization of a token with respect to the document it belongs to
- Intuition: If token $w$ appears a lot in $d_i$ but rarely in $d_j$, $j \neq i$, then $w$ is important for $d_i$
- How can we express this mathematically?
- We simply take the log of the inverse document frequency:

$$idf(w, D) = log\left(\frac{|D|}{\sum_{d \in D} \mathbb{1}\{w \in d\}}\right) \quad (1)$$

# Let's put it together: tf-idf

- Let's put the notion of *tf* and *idf* together
- Unsurprisingly, we will refer to the whole transformation as *tf-idf*
- Interpretation: $\textit{tf-idf}(w, d, D)$ tell us the importance of token $w$ in document $d$ given its usage in $D$.
- We calculate this quantity simply by multiplying *tf* times *idf*:

$$tf - idf(w, d, D) = tf(w, d)idf(w, D) \tag{2}$$

⇒ This is how early search engines performed queries!

# Cosine Similarity

- Now let's take a step back and think about what we have constructed
- For each $d \in D$ we have a (meaningful) vector representation
- We can now use linear algebra to assess whether two vectors are similar
- We simply check whether they point in a similar direction by calculating the (cosine) of the angle between them: **Cosine Similarity**

$$cos\theta = \frac{A \cdot B}{|A||B|} \tag{3}$$

# Cosine Similarity

- Insert pic of cosine sim
- Cosine Similarity has nice properties:
  1. $cos\theta \in [-1, 1]$
  2. $cos\theta = 1 \Rightarrow v_i = v_j$
  3. $cos\theta = -1 \Rightarrow v_i, v_j$ point in opposite directions
  4. $cos\theta$ increases in similarity
- Let's look at some code!