Document Classification

- Document -1 vector: d1$=(x_1, \ldots x_{\mathcal{N}}),\ x_i \in \mathbb{R}$
- Document -2 vector: d2$=(y_1, \ldots y_{\mathcal{N}}),\ y_i \in \mathbb{R}$
- Design a Similarity function: f(d1, d2)
  - A good document similar function should identify relevant documents
- Similarity function based on
  - Bag of Words
  - Term Frequency (TF)
  - Document Length

Many Similarity Function

- Distance Function (q, d)
  - **Vector space model**
  - Probabilistic models P(s=1 | q, d)
  - Probabilistic inference model f(q, d) = p(d→q)
  - Axiomatic model
  - Deep Learning

Vector Space Model (VSM) Assumptions

- Terms are assumed to be orthogonal (independent from each other)
- Term: basic concepts such as word or phrase (n-gram)
- N terms define an N-dimensional space
- Vector Placements base on **Terms** $(w_1, \ldots w_N)$
- Term weight in query and document indicates how well the term characterizes the document or query.

We need to define a similarity function *Distance Function (q, d)* for measuring the relevance.

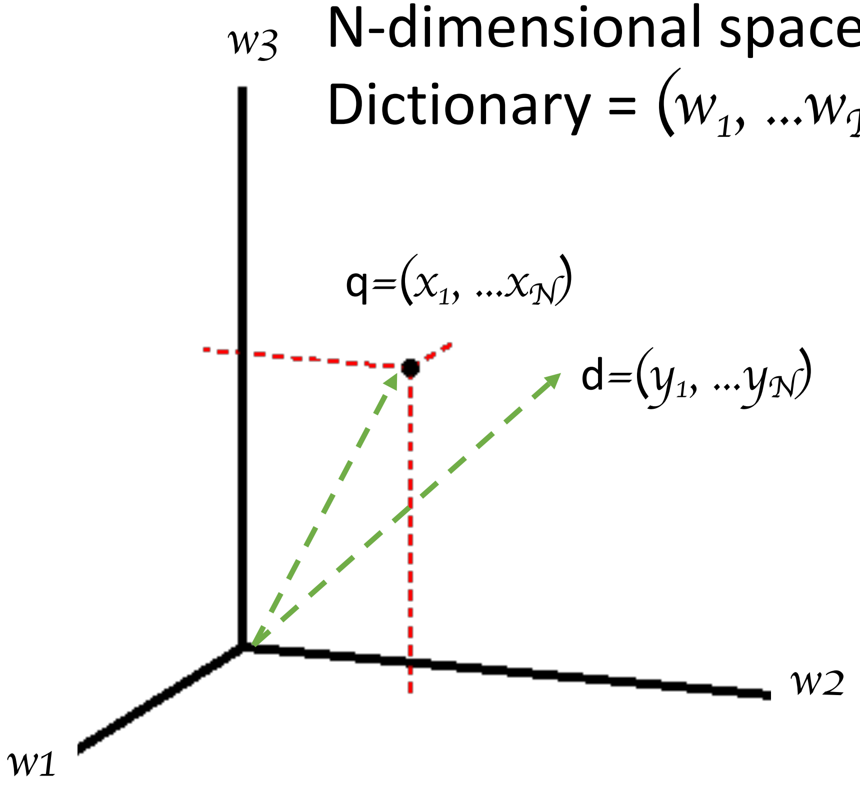
Bit Vector Presentation

- Represent a document or query by a **term vector**
- Query vector: q=$(x_1, \ldots x_{\mathcal{N}})$, $x_i \in \{0,1\}$ *is query term weight*
- Document vector: d=$(y_1, \ldots y_{\mathcal{N}})$, $y_i \in \{0,1\}$ *is document term weight*

1: if word w_i is present

0: if word w_i is not present

N-dimensional space, Dictionary = $(w_1, \ldots w_{\mathcal{N}})$



Similarity → Distance Function: DoT Product

Dictionary = $(w_1, \ldots w_{\mathcal{N}})$
q= $(x_1, \ldots x_{\mathcal{N}})$, $x_i \in \{0,1\}$
d= $(y_1, \ldots y_{\mathcal{N}})$, $y_i \in \{0,1\}$

IF word w_i is present in q or d THEN $x_i = 1$
IF word w_i is absent in q or d THEN $x_i = 0$

$$\mathbf{D\ (q, d)} = \vec{q} \cdot \vec{d} = x_1 y_1 + x_2 y_2 + \ldots\ x_n y_n = \sum_{i=1}^{N} x_i y_i$$

Example 1

- I hate running
- I like NLP
- I like deep learning

Terms = {I, like, hate, Deep, Learning, NLP, flying}

Example 2

Query = news about sport campaign

Document 1 = .. news <u>about</u> food campaign

Document 2 = .. news of sport campaign

Document 3 = .. news of <u>sport</u> campaign … <u>sport</u> activities ..

Similarity → Distance Function: DoT Product

Improved SVM with **Term Frequency Weighting**

Dictionary = $(w_1, \ldots w_{\mathcal{N}})$

q= $(x_1, \ldots x_{\mathcal{N}})$, $x_i = count\ of\ word\ wi\ in\ q = c(w_i, q)$

d= $(y_1, \ldots y_{\mathcal{N}})$, $y_i = count\ of\ word\ wi\ in\ d = c(w_i, d)$

D (q, d) = $\vec{\boldsymbol{q}} \cdot \vec{\boldsymbol{d}} = x_1y_1 + x_2y_2 + \ldots.\ x_ny_n = \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} c(w,q)c(w,d)$

IDF Weighting
Penalizing popular terms

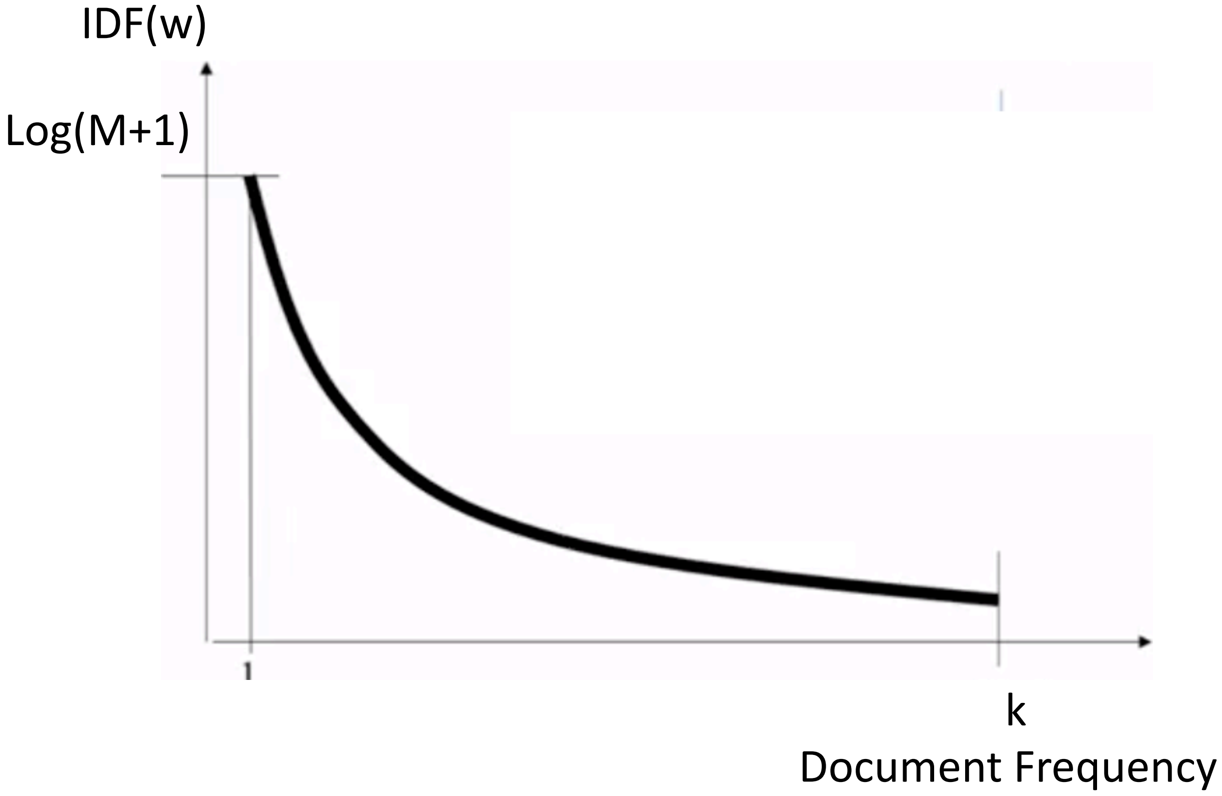
$$\textbf{IDF (w)} = \log\frac{M+1}{\mathbf{k}(w)}$$

$\boldsymbol{M}$ = total number of docs in collection

$\boldsymbol{df}(\boldsymbol{w})$ = total number of docs containing w
(Document frequency)

k(w) < M

k(w) → M and M is large IDF(w) → 0



Similarity → Distance Function: DoT Product

Further Improvement of Vector Placement

Inverse Document Frequency (IDF)

Dictionary = $(w_1, \ldots w_{\mathcal{N}})$

q= $(x_1, \ldots x_{\mathcal{N}})$, $x_i = count\ of\ word\ wi\ in\ q = c(w, q)$

d= $(y_1, \ldots y_{\mathcal{N}})$, $y_i = count\ of\ word\ wi\ in\ d$ * $\mathcal{IDF}\ (wi)$

= $c(w, d)$ * $\mathcal{IDF}\ (w_i)$

D (q, d) = $\vec{q} \cdot \vec{d} = x_1y_1 + x_2y_2 + \ldots x_ny_n$

$$= \sum_{i=1}^{N} c(w, q) * c(w, d) * IDF(w)$$

Similarity Function with TF-IDF Weighting

$$f(q,d) = \sum_{i=1}^{N} x_i y_i = \sum \underbrace{c(w,q)c(w,d)}_{\text{TF(w)}} \underbrace{\log\frac{M+1}{df(w)}}_{\text{IDF(w)}}$$

$Query: \boldsymbol{q}$
$Documnent: \boldsymbol{d}$
All matches query words in d: $\boldsymbol{w} \in \boldsymbol{q} \cap \boldsymbol{d}$
Document Frequency: df$(\boldsymbol{w})$
Tool number of documents in collection: M
Count of word w in query q: $\boldsymbol{c}(\boldsymbol{w}, \boldsymbol{q})$
Count of word w in document d: $\boldsymbol{c}(\boldsymbol{w}, \boldsymbol{d})$

Example 2

Query = news about sport campaign

Document 1 = .. news about food campaign

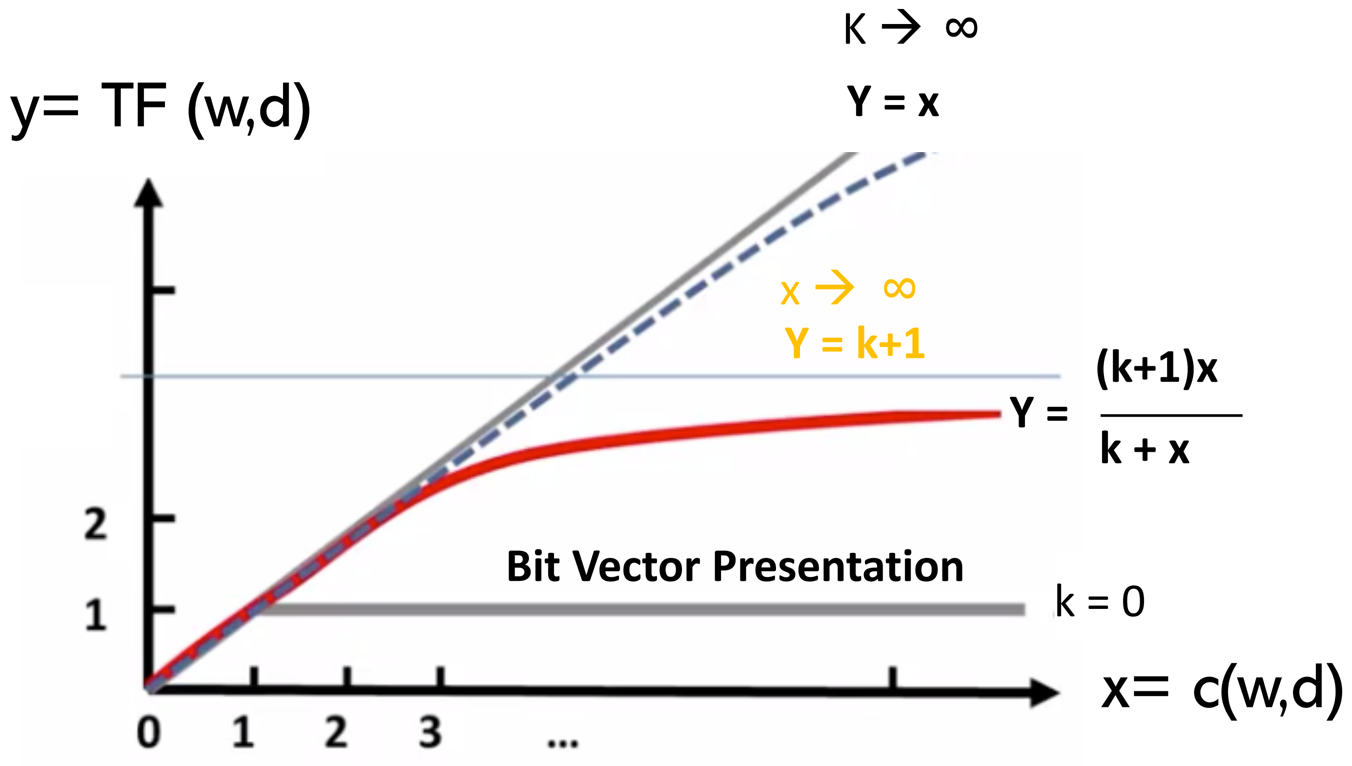
Document 2 = .. news of sport campaign

Document 3 = .. news of sport campaign … sport activities ..

Document 4 = .. news of food campaign … campaign … campaign

$$\mathrm{f}(q, doc\ 4) = \sum_{i=1}^{N} x_i y_i = \sum c(w,q)c(w,d) \underbrace{\log\frac{M+1}{df(w)}}_{\text{c ("campaign", doc4)} \,*\, \log\frac{5}{4} =}$$

BM25 Transformation



BM25 Transformation

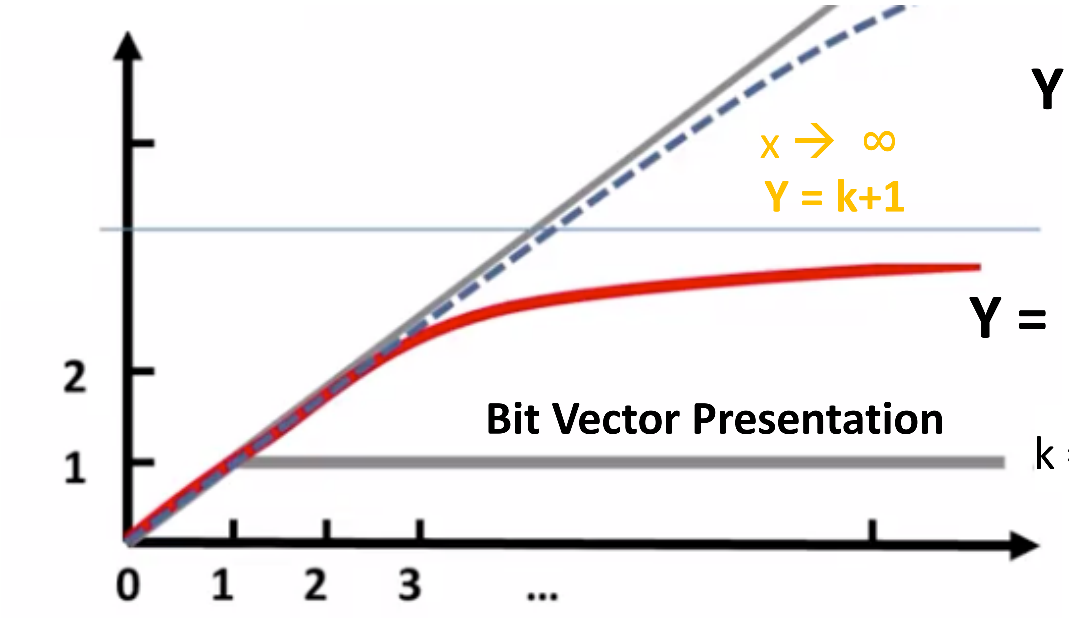
$$f(q,d) = \sum_{i=1}^{N} x_i y_i = \sum c(w,q)c(w,d)\frac{(k+1)}{k+c(w,d)}\log\frac{M+1}{df(w)}$$

K → ∞

Y = x

k >=0

y= TF (w,d)



Y = ln (1+ ln(1 + c(w,d)))

$$Y = \frac{(k+1)\,c(w,d)}{k + c(w,d)}$$

k = 0

x= c(w,d)

Document Length Normalization

A long document has a higher chance to match any query so we should penalize the document length

$$\text{Normalizer} = 1\text{-}b\text{+}b\ \frac{d}{Avg(all\ d)}$$

b is [0, 1]

BM25/Okapi

A long document has a higher chance to match any query so we should penalize the document length

Normalizer = 1-b+b $\frac{d}{Avg(all\ d)}$

b is [0, 1]

$$f(q,d) = \sum_{i=1}^{N} x_i y_i = \sum c(w,q)c(w,d)\frac{(k+1)}{k+c(w,d)}\frac{1}{1-b+\mathrm{b}\frac{d}{Avg(all\ d)}}\log\frac{M+1}{df(w)}$$

Pivoted Length Normalization VSM

$$f(q,d) = \sum_{i=1}^{N} x_i y_i = \sum c(w,q) \frac{\ln(1+\ln(1+c(w,d)))}{1-b+\mathrm{b}\,\frac{d}{Avg(all\ d)}} \log \frac{M+1}{df(w)}$$