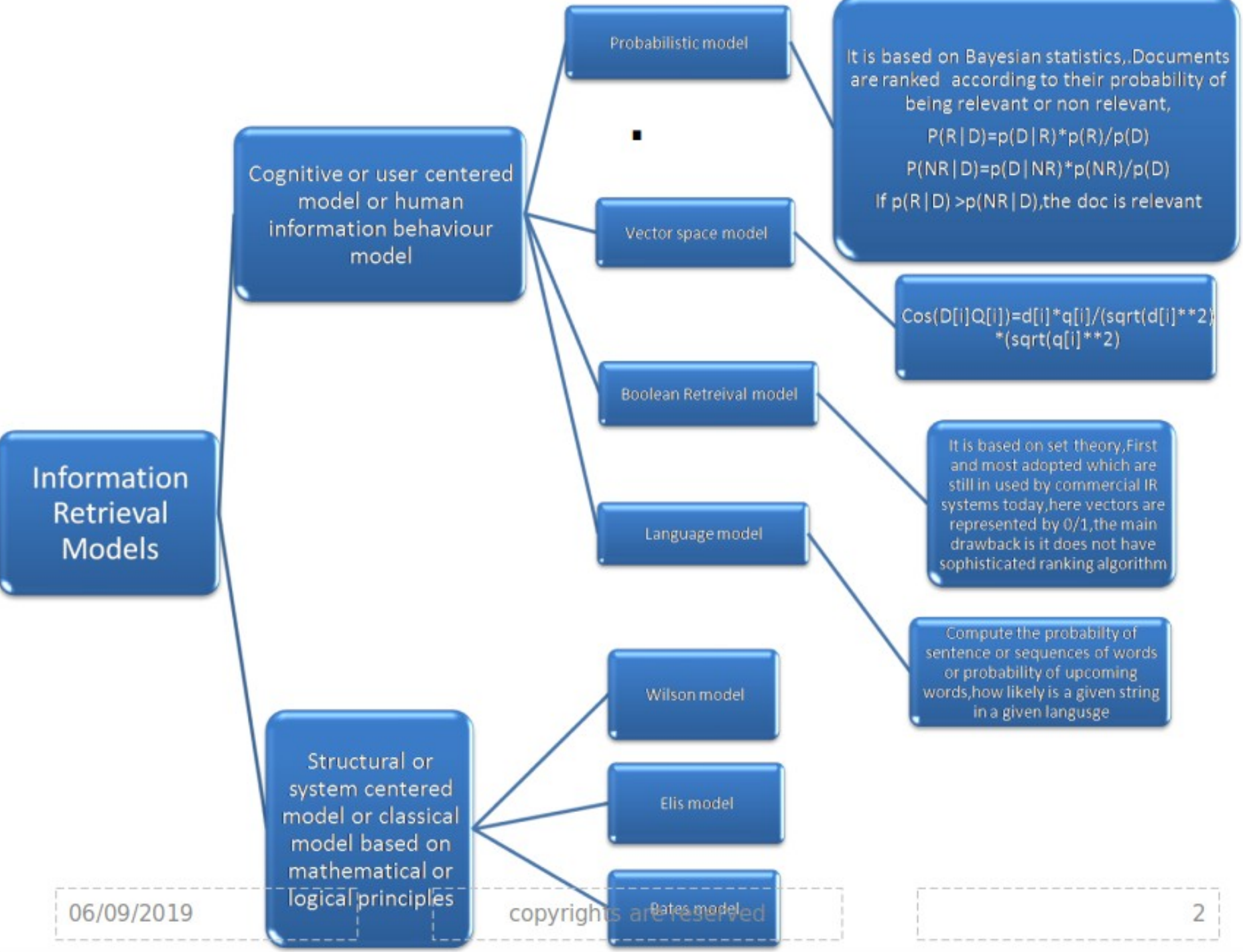
VECTOR SPACE MODEL

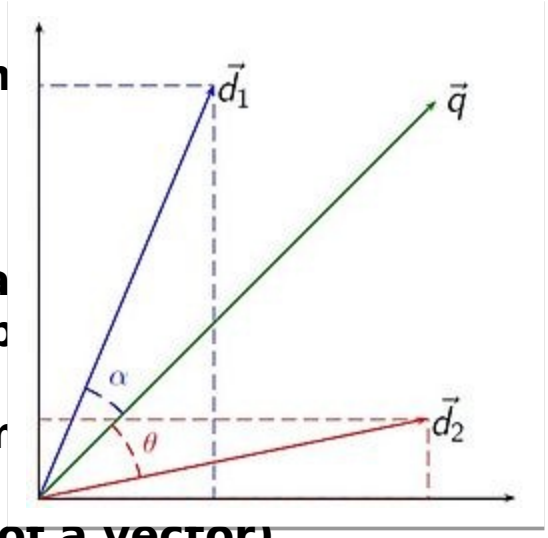
Submitted by PAULAMI DAS

IBAB,BANGALORE



WHAT IS VECTOR SPACE MODEL:

- **Another name is Term Vector Model or Vector Processing**
- **It represents both docs and queries by term sets and compare global similarities between query and documents**
- **It was first used by SMART information Retrieval System**
- **Used to compare different texts and relevant records similar to the queries**
- **Terms are single words/key words or phrases**
- **The dimentionality of the vector is the no of words in**
- **A text can be represented by the words it contains**
- **Bag of Words representation**
- **It is used in NLP,information retrieval and Document**
- **It is used to determine which doc is more similar to a**
- **Documents and queries are represented in a same sp**
- **Formula's:**



i and j are two documents ,k – term and t- last term

$$\sum_{k=1}^{T} Term[ik]$$ **(the sum of weight of all properties of a vector)**

$$\sum_{k=1}^{T} Term[ik].Term[jk]$$ **(the sum of product of term weights for two vector)**

$$\sum_{k=1}^{T} \min(Term[ik],Term[jk])$$ **(minimum of the sum of prodt of term weights for two vector**

Similarity coefficient

Similarity Measure Sim (X,Y)	Binary Term Vectors	Weighted Term Vectors
Dice coefficient	$2\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$	$\frac{2\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2}$
Cosine coefficient	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert^{1/2} \lvert Y \rvert^{1/2}}$	$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{t} x_i^2 \cdot \sum_{i=1}^{t} y_i^2}}$
Jaccard coefficient	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert - \lvert X \cap Y \rvert}$	$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2 - \sum_{i=1}^{t} x_i \cdot y_i}$

Calculation of Dice Coefficient

- **Sim(doci,docj)=**
- **2 x [∑Tk=1 Termik*Termjk)]/T∑k=1Termik+T∑k=1Termjk**
- **Doci=[3,2,10,0,0,1,1]**
- **Docj=[1,1,1,0,01,0,0]**
- **2x[(3x1)+(2x1)+(1x1)+(0x0)+(0x0)+(0x1)+(1x0)+(1x0)]/ (3+2+1+0+0+0+1+1)+(1+1+1+0+0+1+0+0)**
  =12/12=1

Calculation of Jacard Coefficient

- **Sim(doci,docj)=**
- **∑Tk=1 (Termik*Termjk)/T∑k=1Termik+T∑k=1Termjk+(T∑k=1Termik-T∑k=1Termjk)**
- **Doci=[3,2,10,0,0,1,1]**
- **Docj=[1,1,1,0,01,0,0]**
- **6/8+4-6 =6/6 =1**

BAG OF WORDS MODEL

It is a puppy .it is cute

Text preprocessing for modelling,Text are generating from different types of source

- **Take full set of documents what you want to quantify**

TOKENIZATION:break out every single element into a corpus,so that we can have a look at them,

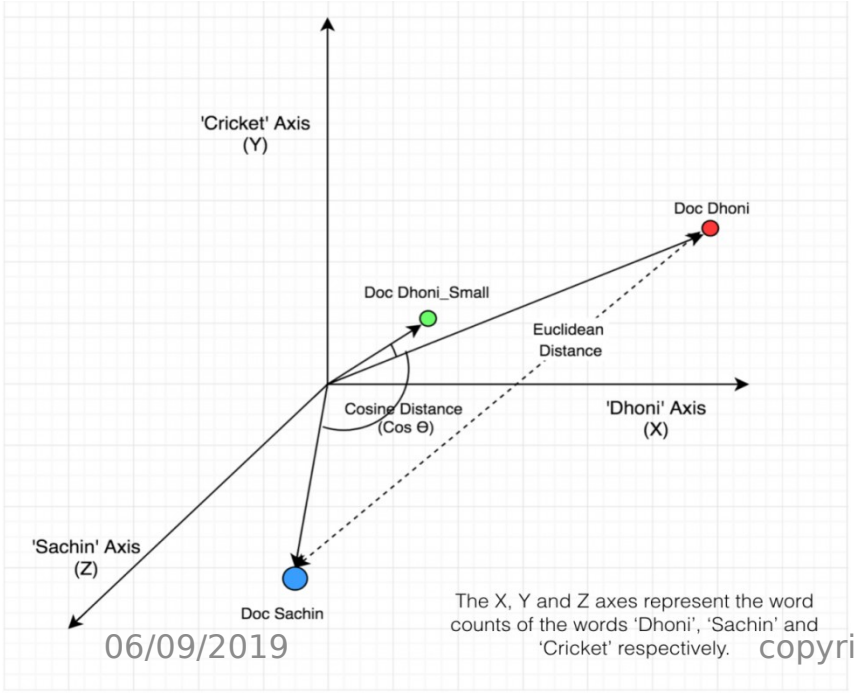
Filtering out the stop words(most common word which does not contain relevant information
Like SO,WHO,OR,A,AN,THE and then take the remaining token

Stem and lemmitizing it to get the root words

word	Freq counts
they	0
puppy	1
and	1
cat	0
good	0
cute	1
…	…

Cosine Similarities

- **To measure the similarities between documents irespecively of their sizes**
- **Mathematically it measures the cosine of the angle between two vectors projecting in a multidimentional space**
- **The angle between two vectors or rather than cosine of that angle is used as a proxy for the similarity of the documents**
- **Smallar angle means larger similarity**
- **As the range of cosα=[-1,1],but text count cant be negetive,so the range is [0,1]**



$$\mathbf{cosim}(A,B) \stackrel{def}{=} \frac{\sum_{i=1}^{n}(A_i * B_i)}{\sqrt{\sum_{i=1}^{n} A_i^2} * \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Contd..

Document 1: IBAB has bigdata course for biology

Document 2: learning biology is easy

Document 3: bigdata is a vast subject

query: bigdata biology

	IBAB	learning	has	bigdata	course	for	biology	is	a	vast	subject	easy
doc 1	1	0	1	1	1	1	1	0	0	0	0	0
doc 2	0	1	0	0	0	0	1	1	0	0	0	1
doc 3	0	0	0	1	0	0	0	1	1	1	1	0
query	0	0	0	1	0	0	1	0	0	0	0	0

Cossim(doc1,query)=((1x1)+(1x1))/2.44 =0.81

Cossim(doc2,query)=1/3.74=0.26

Cossim(doc3,query)=1/3.74=0.26 as we see doc1 is our relevant docs

Tf-idf Model

- **Term Frequency also known (TF) measures the number of times word w occurs in a document/total number of words in a documents.**
- **Inverse Document Frequency (IDF) measures log (no of documents /no of docs that contaains word w)**
- **In information retrieval, tf–idf or TFIDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.**
- **It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling.**
- **tf–idf is one of the most popular termweighting schemes today; 83% of text-based recommender systems in digital libraries use tf–idf.**
- **Variations of the tf–idf weighting scheme are often used by search engines(google) as a central tool in scoring and ranking a document's relevance given a user query.**
- **tf–idf can be successfully used for stop-words filtering in various subject fields, including text summarization and classification.**
- **One of the simplest ranking functions(page rank algorithm)**
- **is computed by summing the tf–idf for each query term.**

$$\mathbf{tf}(t,d) = \frac{f_d(t)}{\max_{w \in d} f_d(w)}$$

$$\mathbf{idf}(t,D) = \ln\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right)$$

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

$$\mathbf{tfidf}'(t,d,D) = \frac{\mathbf{idf}(t,D)}{|D|} + \mathbf{tfidf}(t,d,I$$

$f_d(t) :=$ frequency of term t in document d

$D :=$ corpus of documents

TF–PDF (Term Frequency * Proportional Document Frequency).

TF–PDF was introduced in 2001 in the context of identifying emerging topics in the media. The PDF component measures the difference of how often a term occurs in different domains.

- **Another derivate is TF–IDuF. In TF–IDuF,idf is not calculated based on the document corpus that is to be searched or recommended. Instead, idf is calculated on users' personal document collections. The authors report that TF–IDuF was equally effective as tf–idf but could also be applied in situations when, e.g., a user modeling system has no access to a global document corpus.**

Contd..

Document 1: IBAB has bigdata course for biology

Document 2: learning biology is easy

Document 3: bigdata is a vast subject

query: bigdata biology

Normalised TF for all the documents: TF=(NO OF TIMES WORD APPEAR IN A DOCS/NO OF WORDS IN A DOCS)

docs	learning	biology	IBAB	has	bigdata	course	for	easy	is	a	subject	vast
Doc 1	0	1/6 =0.16	1/6 =0.16	1/6 =0.16	1/6 =0.16	1/6 =0.16	1/6 =0.16	0	0	0	0	0
Doc 2	¼= 0.25	¼= 0.25	0	0	0	0	0	¼= 0.25	¼= 0.25	0	0	0
Doc 3	0	0	0	0	1/5 =0.2	0	0	0	1/5 =0.2	1/5 =0.2	1/5 =0.2	1/5 =0.2

IDF: log(no of docs/no of docs that contains query word)

learning	biology	IBAB	has	bigdata	course	for	easy	is	a	subject	vast
Log(3/1)=0.477	log(3/2)=0.17	Log(3/1)=0.477	Log(3/1)=0.477	Log(3/2)=0.17	Log(3/1)=0.477	Log(3/1)=0.477	Log(3/1)=0.477	Log(3/2)=0.17	Log(3/1)=0.477	Log(3/1)=0.477	Log(3/1)=0.477

Calculation of TF x IDF ,to fond out relevant doc for the query: bigdata biology:

query	doc1	doc2	doc3
bigdata	0.17*0.16=0.0272	0.17*0=0	0.17*0=0
biology	0.17*0.16	0.17*0=0	0.17*0=0

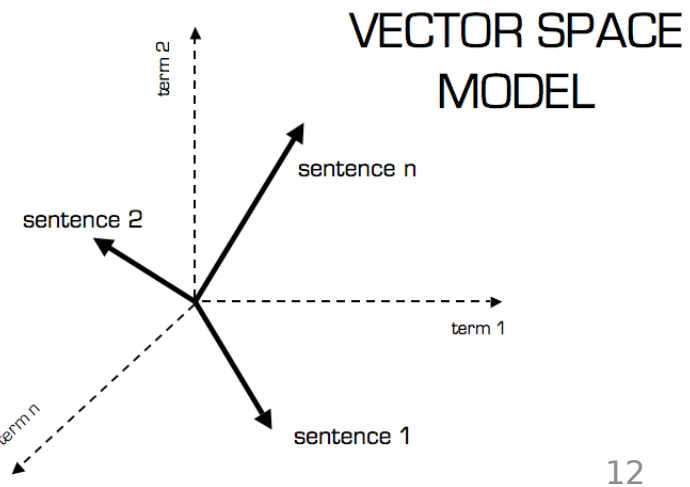
Thus we can say that doc1 is relevant to the query

Pros and Cons of VSM:

- **Pros.**
  - **Allows documents with partial match to be also identified**
  - **The cosine formula gives u a score which can be used to order documents**
  - **If two documents are far apart by Euclidean distance due to huge size of a particular documents, Chances are still there as they may still be oriented closer together , as it captures the orientation of the documents not the magnitude**
- **Cons**
  - **As documents are treated as Bag of words model so no positional information will be there**
  - **It lacks the guidance of details of how weighting and ranking algo are related to relevance**

Application of VSM in BioInformatics

- **DNA sequence searching**
- **Particular gene which is highly up regulated or downregulated in some RNA sequence (by counting the frequency of that gene sequences)**
- **PSSM**
- **Sequence encoding using term document matrix**
- **Clustering**
- **Database search**
- **Pattern classification**
- **Sequence alignment**
- **Finding consensus sequence**



Implementation in python

- https://github.com/PAUL-8274/seminar_on_vector_space_model

Reference:

- www.machinelearningplus.com
- Vector Space Information Retrieval Techniques for Bioinformatics Data Mining Eric Sakk and Iyanuoluwa E. Odebode Department of Computer Science, Morgan State University, Baltimore, MD USA
- https://janav.wordpress.com/2013/10/27/tf-idf-and-cosine-similarity/
- https://pdfs.semanticscholar.org/eb1e/5b5e264e34b0fdd786b064208581e36c6cc5.pdf
- http://bscit.berkeley.edu/cgi-bin/pl_dochome?query_src=&format=html&collection=Wilensky_papers&id=3&show_doc=yes
- http://lucene.apache.org/core/3_6_1/api/all/org/apache/lucene/search/Similarity.html
- http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer
- https://www.opinosis-analytics.com/knowledge-base/what-is-term-frequency/

THANK YOU