lexicon => 사전(해시 테이블) => 단어:위치, 단어:위치, ...

Posting => 문서:빈도:다음위치, 문서:빈도:다음위치, ...

Local Indexing => Merge(위치를 조정, linked list)

collection(문서집합), Lexicon(사전)

In [1]:

```
collection = [
    ("Document1", "This is a a a a a a a a sample."),
    ("Document2", "This is another sample."),
    ("Document3", "This is not a sample.")
]

globalLexicon = dict() #단어:위치 <- 튜플 no
globalDocument = list()
globalPosting = list()
globalMaxTF = dict()
globalTotalTF = dict()
```

TDM doc1 doc2 doc3 ... => term1 1 0 2 ... term2 term3

Lexicon Posting 단어1:위치 (문서 2의 위치) index(문서1):빈도:위치 -1로 <- index(문서 2):빈도:위 치(문서1의)

단어2:위치 index(문서1):빈도:위치 단어3:위치 index(문서1):빈도:위치

document(안 바뀜) 문서1 문서2 문서3

In [12]:

```
#엘라스틱 서치 , 루쉰
```

In [2]:

```
for docName, docContent in collection:
    docIdx = len(globalDocument)
    globalDocument.append(docName)
    #print(globalDocument.index(docName))
    localPosting = dict() #단어:위치
    for term in docContent.lower().split():
        if term in localPosting.keys():
            localPosting[term] += 1
        else:
            localPosting[term] = 1
    globalMaxTF[docIdx] = max(localPosting.values())
    globalTotalTF[docIdx] = sum(localPosting.values())
### Local end ###
### skip sorting ###
    for term, freq in localPosting.items():
        if term in globalLexicon.keys():
            termIdx = list(globalLexicon.keys()).index(term)
            postingIdx = len(globalPosting)
            globalPosting.append((docIdx, freq, globalLexicon[term]))
            globalLexicon[term] = postingIdx
        #Lexicon term ? => 기록, 위치를 업데이트
        else:
            termIdx = len(globalLexicon.keys())
            postingIdx = len(globalPosting)
            globalLexicon[term] = postingIdx
            globalPosting.append((docIdx, freq, -1))
        #Posting 기록, 위치도 기록
```

In [3]:

```
print(globalLexicon)
print(globalPosting[8], globalPosting[4], globalPosting[0])
print(globalDocument[2], globalDocument[1], globalDocument[0])
```

```
{'this': 8, 'is': 9, 'a': 11, 'sample.': 12, 'another': 6, 'not': 10}
(2, 1, 4) (1, 1, 0) (0, 1, -1)
Document3 Document2 Document1
```

In [4]:

```
for term, postingIdx in globalLexicon.items():
    print(term)

    while True:
        if postingIdx == -1:
            break
        print(" {0}-{1}, {2}, Next={3}".format(
            globalPosting[postingIdx][0],
            globalDocument[globalPosting[postingIdx][0]],
            globalPosting[postingIdx][1],
            globalPosting[postingIdx][2]))
        postingIdx = globalPosting[postingIdx][2]
```

```
this
 2-Document3, 1, Next=4
 1-Document2, 1, Next=0
 0-Document1, 1, Next=-1
is
 2-Document3, 1, Next=5
 1-Document2, 1, Next=1
 0-Document1, 1, Next=-1
a
 2-Document3, 1, Next=2
 0-Document1, 8, Next=-1
sample.
 2-Document3, 1, Next=7
 1-Document2, 1, Next=3
 0-Document1, 1, Next=-1
another
 1-Document2, 1, Next=-1
not
 2-Document3, 1, Next=-1
```

In [ ]:

```
#Ranking
```

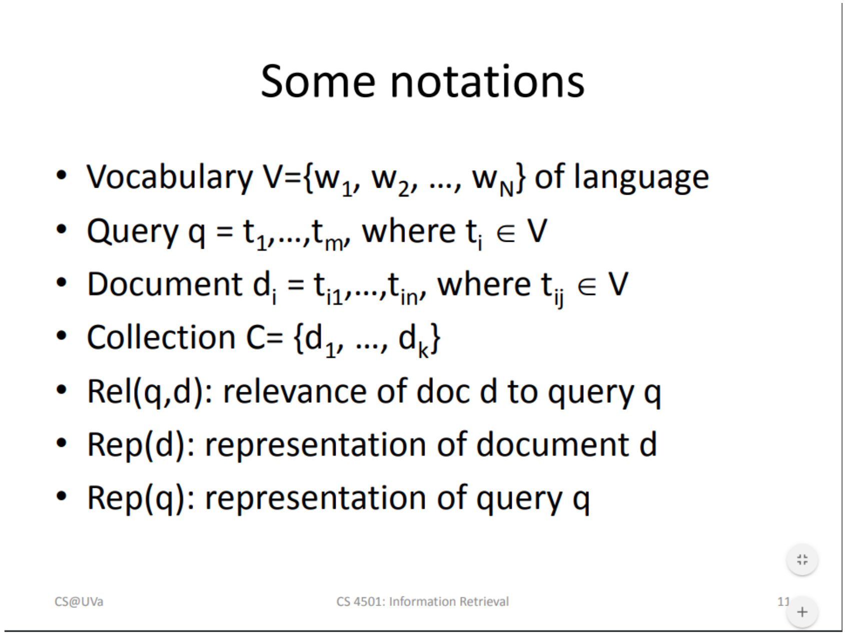
http://www.cs.virginia.edu/~hw5x/Course/IR2015/_site/docs/PDFs/Boolean&VS%20model.pdf (http://www.cs.virginia.edu/~hw5x/Course/IR2015/_site/docs/PDFs/Boolean&VS%20model.pdf)

Vector space model

In [38]:

```
http://www.cs.virginia.edu/~hw5x/Course/IR2015/_site/docs/PDFs/Boolean&VS%20model.pdf
```

Out[38]:



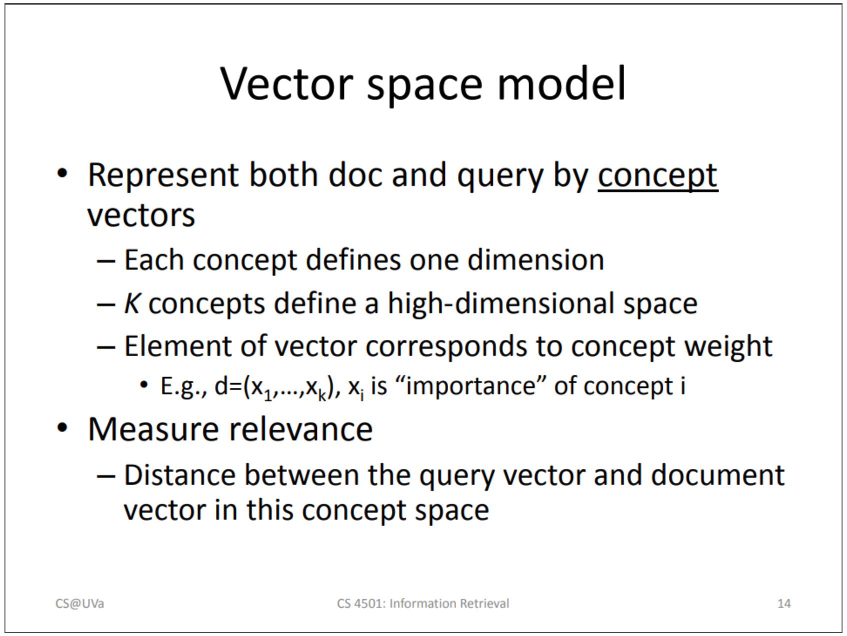
Some notations

- Vocabulary $V=\{w_1, w_2, \ldots, w_N\}$ of language
- Query $q = t_1,\ldots,t_m$, where $t_i \in V$
- Document $d_i = t_{i1},\ldots,t_{in}$, where $t_{ij} \in V$
- Collection $C= \{d_1, \ldots, d_k\}$
- Rel(q,d): relevance of doc d to query q
- Rep(d): representation of document d
- Rep(q): representation of query q

CS@UVa CS 4501: Information Retrieval 11

In [39]:

Out[39]:



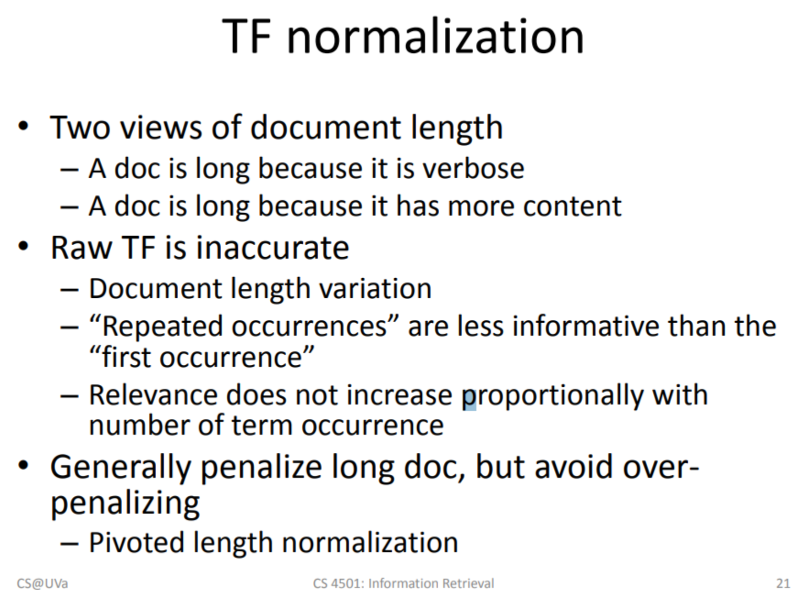
Vector space model

- Represent both doc and query by concept vectors
  - Each concept defines one dimension
  - *K* concepts define a high-dimensional space
  - Element of vector corresponds to concept weight
    - E.g., $d=(x_1,\ldots,x_k)$, x_i is “importance” of concept i
- Measure relevance
  - Distance between the query vector and document vector in this concept space

CS@UVa CS 4501: Information Retrieval 14

In [40]:

Out[40]:



TF normalization

- Two views of document length
  - A doc is long because it is verbose
  - A doc is long because it has more content
- Raw TF is inaccurate
  - Document length variation
  - “Repeated occurrences” are less informative than the “first occurrence”
  - Relevance does not increase proportionally with number of term occurrence
- Generally penalize long doc, but avoid over-penalizing
  - Pivoted length normalization

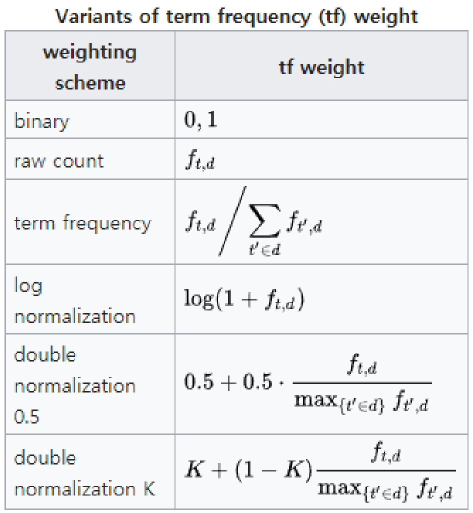
CS@UVa CS 4501: Information Retrieval 21

TF
특정 문서 내 특정 단어의 빈도 = f(t,d)
특정 문서 내 다른 단어의 빈도 => sum, max <- DTM(sum, max) 사용 가능 (*)

In [41]:

```
https://en.wikipedia.org/wiki/Tf%E2%80%93idf
```

Out[41]:



Variants of term frequency (tf) weight

weighting scheme	tf weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Bigg/ \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K)\frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

In [5]:

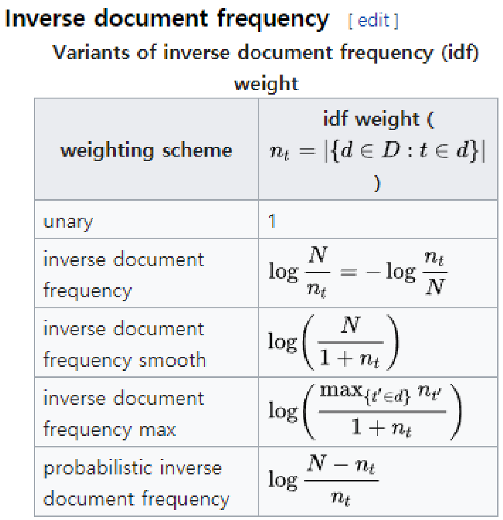
```
from math import log10 #스케일(크기)에 따라 자유선택

def logTF(freq):
    return log10(1+freq)

def totalTF(freq, totalFreq):
    return freq / totalFreq

#가장 많이 활용
def doubleTF(freq, maxFreq, alpha=0.5):
    return alpha+(1-alpha)*(freq / maxFreq)
```

IDF
전체 문서의 개수 => N   <- len(collection)
특정 단어가 나타난 문서의 개수 => df(t)
is a the of => stopwords = 0
not 처리 <- ex) to be or not to be 가 0 처리 -> 데이터 보고 판단

In [42]:

```
https://en.wikipedia.org/wiki/Tf%E2%80%93idf
```

Out[42]:



Inverse document frequency [edit]

Variants of inverse document frequency (idf) weight

weighting scheme	idf weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log\left(\frac{N}{1 + n_t}\right)$
inverse document frequency max	$\log\left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

In [6]:

```
#가장 많이 활용
def idf(df, N):
    return log10(N / df)

#차선
def smoothingIdf(df, N):
    return log10(N / (1+df))

def probabilisticIdf(df, N):
    return log10((N-df+1) / df)
```

In [7]:

```
N = len(globalDocument)

for term, postingIdx in globalLexicon.items():
    print(term)

    df = 0
    while True:
        if postingIdx == -1:
            break
        df += 1
        postingIdx = globalPosting[postingIdx][2]
    print("{0} - DF1:{1}".format(term, idf(df, N)))
    print("{0} - DF2:{1}".format(term, smoothingIdf(df, N)))
    print("{0} - DF3:{1}".format(term, probabilisticIdf(df, N)))
    print()
```

```
this
this - DF1:0.0
this - DF2:-0.12493873660829995
this - DF3:-0.47712125471966244

is
is - DF1:0.0
is - DF2:-0.12493873660829995
is - DF3:-0.47712125471966244

a
a - DF1:0.17609125905568124
a - DF2:0.0
a - DF3:0.0

sample.
sample. - DF1:0.0
sample. - DF2:-0.12493873660829995
sample. - DF3:-0.47712125471966244

another
another - DF1:0.47712125471966244
another - DF2:0.17609125905568124
another - DF3:0.47712125471966244

not
not - DF1:0.47712125471966244
not - DF2:0.17609125905568124
not - DF3:0.47712125471966244
```

In [8]:

```
N = len(globalDocument)

for term, postingIdx in globalLexicon.items():
    df = 0
    while True:
        if postingIdx == -1:
            break
        df += 1
        postingIdx = globalPosting[postingIdx][2]
    print("{0} - DF1:{1}, N:{2}".format(term, df, N))
    print("IDF1:{1}".format(term, idf(df, N)))
    print("IDF2:{1}".format(term, smoothingIdf(df, N)))
    print("IDF3:{1}".format(term, probabilisticIdf(df, N)))
    print()
```

```
this - DF1:3, N:3
IDF1:0.0
IDF2:-0.12493873660829995
IDF3:-0.47712125471966244

is - DF1:3, N:3
IDF1:0.0
IDF2:-0.12493873660829995
IDF3:-0.47712125471966244

a - DF1:2, N:3
IDF1:0.17609125905568124
IDF2:0.0
IDF3:0.0

sample. - DF1:3, N:3
IDF1:0.0
IDF2:-0.12493873660829995
IDF3:-0.47712125471966244

another - DF1:1, N:3
IDF1:0.47712125471966244
IDF2:0.17609125905568124
IDF3:0.47712125471966244

not - DF1:1, N:3
IDF1:0.47712125471966244
IDF2:0.17609125905568124
IDF3:0.47712125471966244
```

In [9]:

```python
N = len(globalDocument)

for term, postingIdx in globalLexicon.items():
    df = 0
    while True:
        if postingIdx == -1:
            break
        df += 1
        print("{0} - DocIdx:{1}, TF:{2}, Max:{3}, Total:{4}".format(term,
        globalPosting[postingIdx][0], globalPosting[postingIdx][1],
        globalMaxTF[globalPosting[postingIdx][0]],
        globalTotalTF[globalPosting[postingIdx][0]]))
        postingIdx = globalPosting[postingIdx][2]
    print("{0} - DF1:{1}, N:{2}".format(term, df, N))
    print("IDF1:{1}".format(term, idf(df, N)))
    print("IDF2:{1}".format(term, smoothingIdf(df, N)))
    print("IDF3:{1}".format(term, probabilisticIdf(df, N)))
    print()
```

```
this - DocIdx:2, TF:1, Max:1, Total:5
this - DocIdx:1, TF:1, Max:1, Total:4
this - DocIdx:0, TF:1, Max:8, Total:11
this - DF1:3, N:3
IDF1:0.0
IDF2:-0.12493873660829995
IDF3:-0.47712125471966244

is - DocIdx:2, TF:1, Max:1, Total:5
is - DocIdx:1, TF:1, Max:1, Total:4
is - DocIdx:0, TF:1, Max:8, Total:11
is - DF1:3, N:3
IDF1:0.0
IDF2:-0.12493873660829995
IDF3:-0.47712125471966244

a - DocIdx:2, TF:1, Max:1, Total:5
a - DocIdx:0, TF:8, Max:8, Total:11
a - DF1:2, N:3
IDF1:0.17609125905568124
IDF2:0.0
IDF3:0.0

sample. - DocIdx:2, TF:1, Max:1, Total:5
sample. - DocIdx:1, TF:1, Max:1, Total:4
sample. - DocIdx:0, TF:1, Max:8, Total:11
sample. - DF1:3, N:3
IDF1:0.0
IDF2:-0.12493873660829995
IDF3:-0.47712125471966244

another - DocIdx:1, TF:1, Max:1, Total:4
another - DF1:1, N:3
IDF1:0.47712125471966244
IDF2:0.17609125905568124
IDF3:0.47712125471966244

not - DocIdx:2, TF:1, Max:1, Total:5
not - DF1:1, N:3
IDF1:0.47712125471966244
IDF2:0.17609125905568124
IDF3:0.47712125471966244
```

In [10]:

```
N = len(globalDocument)

for term, postingIdx in globalLexicon.items():
    old = postingIdx
    df = 0
    while True:
        if postingIdx == -1:
            break
        df += 1
        postingIdx = globalPosting[postingIdx][2]

    postingIdx = old
    idf1 = idf(df, N)
    idf2 = smoothingIdf(df, N)
    idf3 = probabilisticIdf(df, N)

    while True:
        if postingIdx == -1:
            break

        data = globalPosting[postingIdx]
        postingIdx = data[2]
        tf = doubleTF(data[1], globalMaxTF[data[0]])
        print("{0}-{1}".format(term, globalDocument[data[0]]))
        print("=> {0} * {1} = {2}".format(tf, idf1, tf *idf1))
        print("=> {0} * {1} = {2}".format(tf, idf2, tf *idf2))
        print("=> {0} * {1} = {2}".format(tf, idf3, tf *idf3))
    print()
```

```
this-Document3
=> 1.0 * 0.0 = 0.0
=> 1.0 * -0.12493873660829995 = -0.12493873660829995
=> 1.0 * -0.47712125471966244 = -0.47712125471966244
this-Document2
=> 1.0 * 0.0 = 0.0
=> 1.0 * -0.12493873660829995 = -0.12493873660829995
=> 1.0 * -0.47712125471966244 = -0.47712125471966244
this-Document1
=> 0.5625 * 0.0 = 0.0
=> 0.5625 * -0.12493873660829995 = -0.07027803934216872
=> 0.5625 * -0.47712125471966244 = -0.2683807057798101

is-Document3
=> 1.0 * 0.0 = 0.0
=> 1.0 * -0.12493873660829995 = -0.12493873660829995
=> 1.0 * -0.47712125471966244 = -0.47712125471966244
is-Document2
=> 1.0 * 0.0 = 0.0
=> 1.0 * -0.12493873660829995 = -0.12493873660829995
=> 1.0 * -0.47712125471966244 = -0.47712125471966244
is-Document1
=> 0.5625 * 0.0 = 0.0
=> 0.5625 * -0.12493873660829995 = -0.07027803934216872
=> 0.5625 * -0.47712125471966244 = -0.2683807057798101

a-Document3
=> 1.0 * 0.17609125905568124 = 0.17609125905568124
=> 1.0 * 0.0 = 0.0
=> 1.0 * 0.0 = 0.0
a-Document1
=> 1.0 * 0.17609125905568124 = 0.17609125905568124
=> 1.0 * 0.0 = 0.0
=> 1.0 * 0.0 = 0.0

sample.-Document3
=> 1.0 * 0.0 = 0.0
=> 1.0 * -0.12493873660829995 = -0.12493873660829995
=> 1.0 * -0.47712125471966244 = -0.47712125471966244
sample.-Document2
=> 1.0 * 0.0 = 0.0
=> 1.0 * -0.12493873660829995 = -0.12493873660829995
=> 1.0 * -0.47712125471966244 = -0.47712125471966244
sample.-Document1
=> 0.5625 * 0.0 = 0.0
=> 0.5625 * -0.12493873660829995 = -0.07027803934216872
=> 0.5625 * -0.47712125471966244 = -0.2683807057798101

another-Document2
=> 1.0 * 0.47712125471966244 = 0.47712125471966244
=> 1.0 * 0.17609125905568124 = 0.17609125905568124
=> 1.0 * 0.47712125471966244 = 0.47712125471966244

not-Document3
=> 1.0 * 0.47712125471966244 = 0.47712125471966244
=> 1.0 * 0.17609125905568124 = 0.17609125905568124
=> 1.0 * 0.47712125471966244 = 0.47712125471966244
```

```
Hosting file
어떤 문서의 인덱스:빈도:다음 문서의 위치   <- TDM length / while을 -1까지

단어:포스팅 위치
```