In [ ]:
```
Preprocessing - Tokeninzing+Normalizing
Tokenizing: N-gram(LM), Stemming(BE, C, BPE), Lemmatization(MA)
Normalizing: 대소문자?, 구두점?, 약어, 고유명사?
             문자열의 길이, 토큰의 빈도(Zipf), 불용어(Stopwords)
                                        사전기반(누군가 관리), BPE(욕설)
             자음+모음, 초중종성=>1음절, 형태소(단어)
             Edit-Distance: Hamming, Lev...(min?)
-----------> 전처리 끝 (토큰 추출/선별 => Feature Extraction/Selection)
AI: AI - ML - DL
TM: Information Retrieval(추천, 분류, 군집, 번역, ...)
문서-토큰 => Encoding => Matrix = ML = DL
```

In [ ]:
```
           추:root
       가  신  ...     천   ---- N
    요     메   3음절..
    ....
BoW(Bag of Words) - Indepedent => Vector Space(Terms)
-> Complexity, Sparse => , Decomposition, Dense(NN-based)
```

In [1]:
```
from os import listdir

def fileids(path):
    fileList = list()

    path = path + ('' if path[-1] == '/' else '/')

    for f in listdir(path):
        if f.endswith('.txt'):
            fileList.append(path+f)

    return fileList
```

In [2]:
```
D = list()
for file in fileids('news'):
    with open(file, 'r', encoding='utf8') as fp:
        D.append(fp.read())
```

In [4]:
```
V = list()
for d in D:
    V.extend(d.split())
V = list(set(V))
```

In [6]:
```
V[:10]
```

Out[6]:
```
['뜷하는',
 '관리사무소',
 '소개자료에',
 '까지',
 '내에서',
 '전망된다"고',
 '먹거리로',
 '음반”이라며',
 '확대와',
 '홈페이지에서는']
```

In [7]:
```
D = [list(set(d.split())) for d in D]
```

In [8]:
```
len(D), len(D[0])
```

Out[8]:
```
(206, 389)
```

In [19]:
```
# Complexity
# Space: len(D) * len(V) - 줄여야함 과제 -> Linked List!
# Time: |Q| * D * |D| - 병목부분 줄여야함 과제 -> Vector Space Model!
Q = '소개자료에 까지 먹거리로'.split() # |Q|=3
# OR 연산 - Boolean 검색
# AND 연산 -
result = list()
for q in Q:
    result.append(list())
    for d in D: # Bottle neck
        for t in d:
            if q == t:
#                 result.append(D.index(d))
                result[-1].append(D.index(d))
                break
# list(set(result))
list(set(result[0]).intersection(set(result[1]))\
     .intersection(set(result[2])))
```

Out[19]:
```
[]
```