# Bag of Words(BoW)

이번 챕터에서는 TDM 행렬이 기본이 되는 개념이자, 단어 등장 순서를 무시하는 빈도수 기반의 방법론인 Bag of Words에 대해서 학습한다.

## 1. Bag of Words란?

Bag of Words란 단어들의 순서는 전혀 고려하지 않고, 단어들의 출현 빈도(frequency)에만 집중하는 텍스트 데이터의 수치화 표현 방법이다. Bag of Words를 직역하면 단어들의 가방이라는 의미이다. 단어들이 들어있는 가방을 상상해보자. 갖고있는 어떤 텍스트 문서에 있는 단어들을 가방에다가 전부 넣는다. 그러고나서 이 가방을 흔들어 단어들을 섞는다. 만약, 해당 문서 내에서 특정 단어가 N번 등장했다면, 이 가방에는 그 특정 단어가 N개 있게 된다. 또한 가방을 흔들어서 단어를 섞었기 때문에 더 이상 **단어의 순서는 중요하지 않는다.**

BoW를 만드는 과정을 이렇게 두 가지 과정으로 생각해본다.

(1) 우선, 각 단어의 고유한 인덱스(Index)를 부여한다.

(2) 각 인덱스의 위치에 단어 토큰의 등장 횟수를 기록한 벡터(Vector)를 만든다.

한국어 예제를 통해서 BoW에 대해서 이해해보자.

이 때문에 모든 sequence의 길이는 동일하다.

문서1 : 정부가 발표하는 물가상승률과 소비자가 느끼는 물가상승률은 다르다.

```
from konlpy.tag import Okt
import re
okt = Okt()


token = re.sub("(\.)","","정부가 발표하는 물가상승률과 소비자가 느끼는 물가
상승률은 다르다.")
# 정규 표현식을 통해 온점을 제거하는 정제 작업이다.
token = okt.morphs(token)
# OKT 형태소 분석기를 통해 토큰화 작업을 수행한 뒤에, token에다가 넣는다.


word2index={}
bow=[]


for voca in token:
    if voca not in word2index.keys():
        word2index[voca] = len(word2index)
        # token을 읽으면서, word2index에 없는 (not in) 단어는 새로 추가하
고, 이미 있는 단어는 넘긴다.
        # 그러면 word2index의 길이이기 때문에 인덱스가 0, 1, 2 .. 이렇게 들
어감
        bow.insert(len(word2index)-1,1)
        # BoW 전체에 전부 기본값 1을 넣어준다. len(word2index)-1은 실제 인
덱스를 의미
        # 단어의 개수는 최소 1개 이상이기 때문이다.
    else:
        index=word2index.get(voca)
        # 재등장하는 단어의 인덱스를 받아온다.
        bow[index] = bow[index]+1
        # 재등장한 단어는 해당하는 인덱스의 위치에 1을 더해준다. (단어의 개수
를 센다)


print(word2index)
{'정부': 0, '가': 1, '발표': 2, '하는': 3, '물가상승률': 4, '과': 5, '소
비자': 6, '느끼는': 7, '은': 8, '다르다': 9}


bow
[1, 2, 1, 1, 2, 1, 1, 1, 1, 1]
```