12. NLP

2019년 4월 16일 화요일 오후 4:48

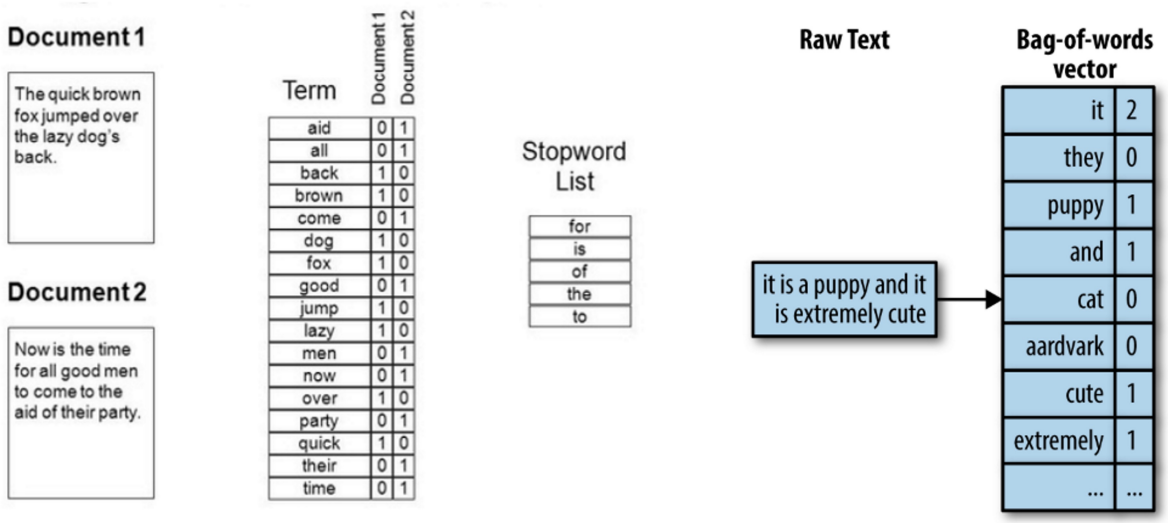
NLT :

- 분류
- 토큰 화
- 형태소 분석
- 태깅
- 구문 분석 및 의미 론적 추론

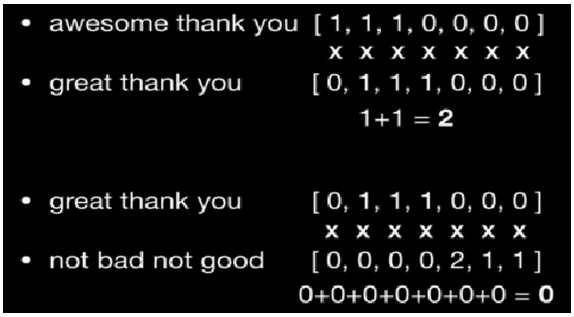
Bag of Words , TF-IDF , Word2vec ,

1. *Bag of Words*

- *단어들의 순서는 전혀 고려하지 않고, 단어들의 출현 빈도(frequency)에만 집중하는 텍스트 데이터의 수치화 표현 방법.*



- **장점**
  a. 문장의 유사도



  b. 머신러닝 input값으로 바로 입력 가능 (텍스트 -> 수)

- **단점**
  a. Sparsity : 데이터의 값이 많이 필요하다.
  b. Frequent words has more power : 오직 빈도에 따라 힘이 강해진다.

c. Ignoring word orders : 단어의 순서를 무시.
d. Out of vocabulary : 오타 , 줄임말을 데이터로 사용하기 어렵다.

2. TF-IDF

- 단어별 문서의 연관성확인
- Term Frequency : 단어가 몇번 출현했는지 빈도가 높을 수 록 연관성이 높다.
- TF score : 단순하게 빈도수로 가중치를 매기기 때문에 요류가 발생 될 확률이 높다.
- IDF score : log ((총 문장) / (특정 단어가 나온 문장))

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF
Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

-

TF-IDF score

- A : "a new car, used car, car review"
- B : "a friend in need is a friend indeed"

word	TF		IDF	TF * IDF	
	A	B		A	B
a	1 / 7	2 / 8	Log (2 / 2) = 0	0	0
new	1 / 7	0	Log (2 / 1) = 0.3	0.04	0
car	3 / 7	0	Log (2 / 1) = 0.3	0.13	0
used	1 / 7	0	Log (2 / 1) = 0.3	0.04	0
review	1 / 7	0	Log (2 / 1) = 0.3	0.04	0
friend	0	2 / 8	Log (2 / 1) = 0.3	0	0.08
in	0	1 / 8	Log (2 / 1) = 0.3	0	0.04
need	0	1 / 8	Log (2 / 1) = 0.3	0	0.04
is	0	1 / 8	Log (2 / 1) = 0.3	0	0.04
indeed	0	1 / 8	Log (2 / 1) = 0.3	0	0.04

3. Word2vec

- One-hot인코딩의 한계 단어간의 유사도를 구할 수 없다.
- Embeding중에 하나

"king brave man"
"queen beautiful woman"

word	neighbor
king	brave
brave	king
brave	man
man	brave
queen	beautiful
beautiful	queen
beautiful	woman
woman	beautiful

"king brave man"
"queen beautiful woman"

word	neighbor
king	brave
king	man
brave	king
brave	man
man	king
man	brave
queen	beautiful
queen	woman
beautiful	queen
beautiful	woman
woman	queen
woman	beautiful

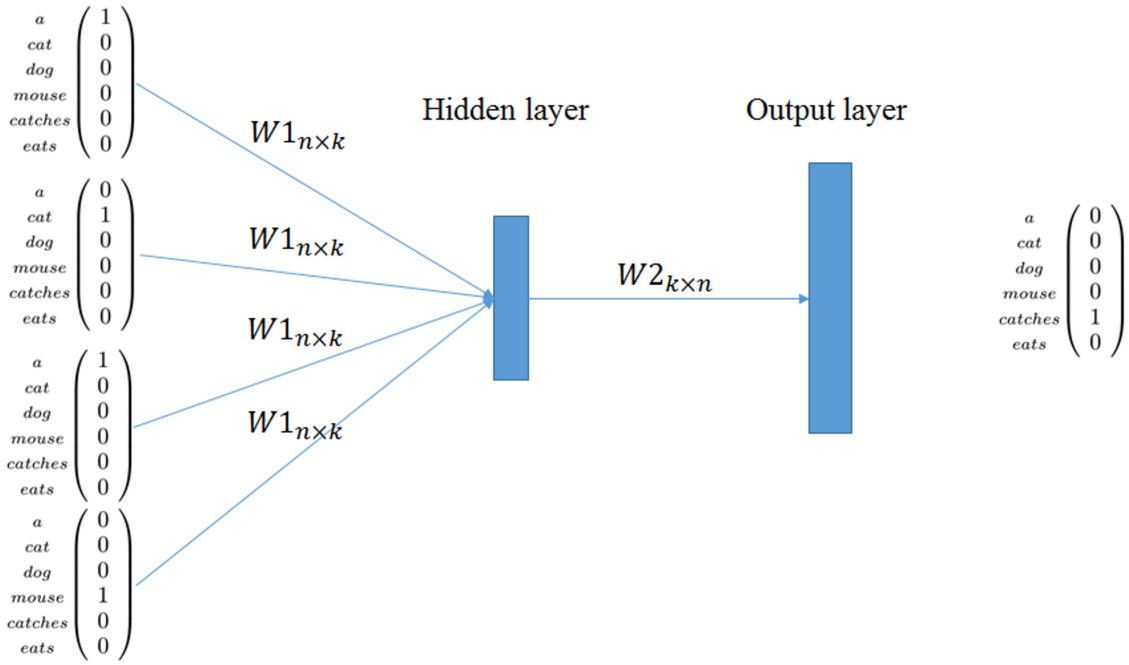
Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

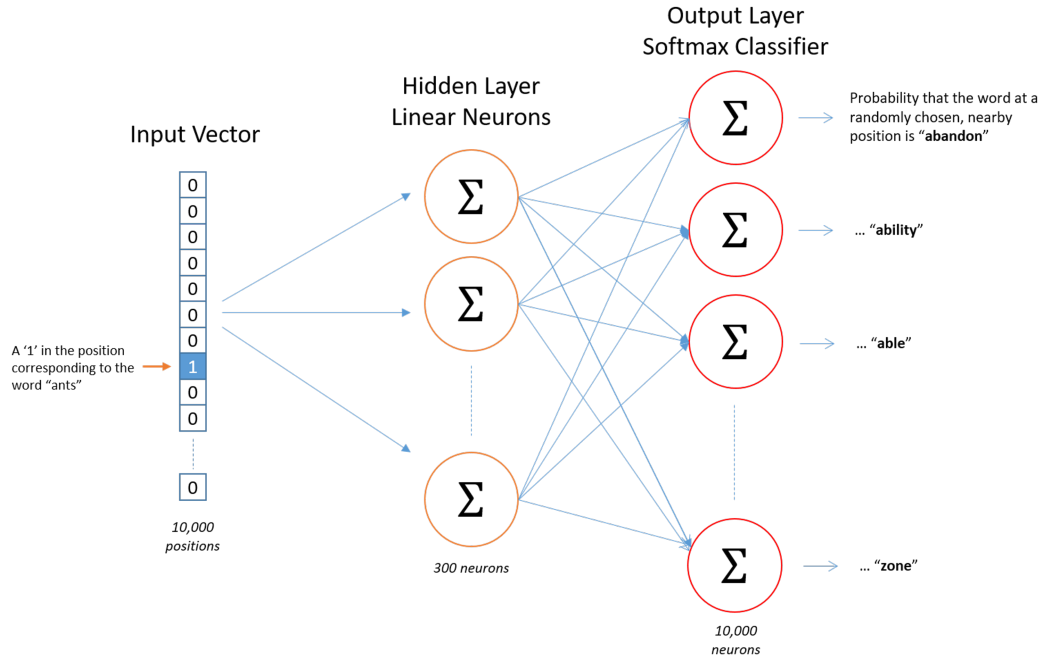
	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Each word gets a 1x9 vector representation

word	word one hot encoding	neighbor	neighbor one hot encoding
king	[1, 0, 0, 0, 0, 0]	brave	[0, 1, 0, 0, 0, 0]
king	[1, 0, 0, 0, 0, 0]	man	[0, 0, 1, 0, 0, 0]
brave	[0, 1, 0, 0, 0, 0]	king	[1, 0, 0, 0, 0, 0]
brave	[0, 1, 0, 0, 0, 0]	man	[0, 0, 1, 0, 0, 0]
man	[0, 0, 1, 0, 0, 0]	king	[1, 0, 0, 0, 0, 0]
man	[0, 0, 1, 0, 0, 0]	brave	[0, 1, 0, 0, 0, 0]
queen	[0, 0, 0, 1, 0, 0]	beautiful	[0, 0, 0, 0, 1, 0]
queen	[0, 0, 0, 1, 0, 0]	woman	[0, 0, 0, 0, 0, 1]
beautiful	[0, 0, 0, 0, 1, 0]	queen	[0, 0, 0, 1, 0, 0]
beautiful	[0, 0, 0, 0, 1, 0]	woman	[0, 0, 0, 0, 0, 1]
woman	[0, 0, 0, 0, 0, 1]	queen	[0, 0, 0, 1, 0, 0]
woman	[0, 0, 0, 0, 0, 1]	beautiful	[0, 0, 0, 0, 1, 0]

input (word one hot encoding)	target (neighbor one hot encoding)
[1, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0]
[1, 0, 0, 0, 0, 0]	[0, 0, 1, 0, 0, 0]
[0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0]	[0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0]	[1, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0]	[0, 1, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0]	[0, 0, 0, 0, 1, 0]
[0, 0, 0, 1, 0, 0]	[0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 1, 0]	[0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 1, 0]	[0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1]	[0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 1]	[0, 0, 0, 0, 1, 0]


a cat dog mouse catches eats
1 0 0 0 0 0
0 1 0 0 0 0
1 0 0 0 0 0
0 0 0 1 0 0
$W1_{n\times k}$
$W1_{n\times k}$
$W1_{n\times k}$
$W1_{n\times k}$
Hidden layer
$W2_{k\times n}$
Output layer
a cat dog mouse catches eats
0 0 0 0 1 0


Output Layer
Softmax Classifier
Hidden Layer
Linear Neurons
Input Vector
A '1' in the position corresponding to the word "ants"
0 0 0 0 0 0 0 1 0 0 0
10,000 positions
300 neurons
10,000 neurons
Probability that the word at a randomly chosen, nearby position is "abandon"
... "ability"
... "able"
... "zone"