텍스트 전처리 (Text Preprocessing)

자연어처리 텍스트마이닝

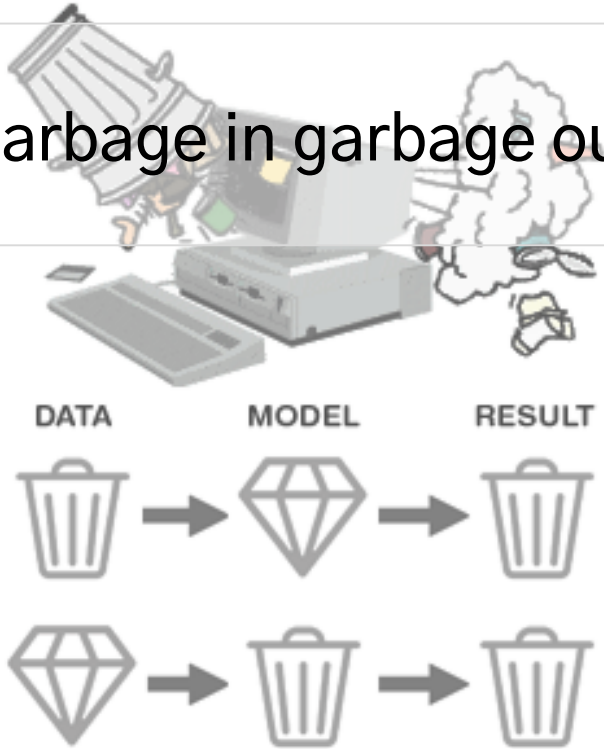
텍스트 전처리 개요

- 분석 하기 전 텍스트를 분석에 적합한 형태로 변환하는 작업
- 전처리 단계는 텍스트를 토큰화하고 자연어 처리에 필요없는 조사, 특수문자, 단어(불용어)의 제거과정을 포함
- 전처리는 분석결과와 모델 성능에 직접 영향을 미치기 때문에 전처리 단계는 매우 중요

1. 토큰화	구두점으로 문서를 문장으로 분리하는 과정이다. 문장부호를 제거하거나 영어의 경우 대문자를 소문자로 변환하는 작업을 할 수 있다.
2. 형태소 분석	뜻을 가진 가장 작은 단위인 형태소로 문장을 분리하는 과정이다.
3. 품사 태깅	분리된 토큰에 품사를 태깅하는 과정이다.
4. 원형 복원	단어 기본 형태인 어간을 추출하는 과정이다. Stemming방식과 Lemmatization방식이 있다.
5. 불용어 처리	분석에 불필요한 단어나 방해되는 단어를 제거하는 과정이다.

텍스트 전처리 개요

garbage in garbage out



토큰화
(Tokenization)

텍스트 전처리 (Text Preprocessing)

토큰화 (Tokenization)

- 텍스트를 자연어 처리를 위해 분리 하는 것
- 토큰화는
  단어별로 분리하는 "단어 토큰화(Word Tokenization)"와
  문장별로 분리하는 "문장 토큰화(Sentence Tokenization)"로 구분

단어 토큰화(Word Tokenization)

- 단어(word)를 기준으로 토큰화
- 영문의 경우 공백을 기준으로 분리하면 유의미한 토큰화가 가능
- 반면 한글의 경우 품사를 고려한 토큰화가 필요

단어 토큰화 고려사항

- 특수문자가 있는 경우

  (구두점 및 특수문자를 단순하게 제외해서는 안됨)

특수문자	원문	토큰화 예제1	토큰화 예제2
‘	Don’t	Do / n’t	Don / ‘ / t
-	State-of-the-art	State / of / the / art	State-of-the-art

- 단어 내 띄어쓰기가 있는 경우

	원문	토큰화 예제1	토큰화 예제2
공백	New York	New / York	New York

문장 토큰화 (Sentence Tokenization)

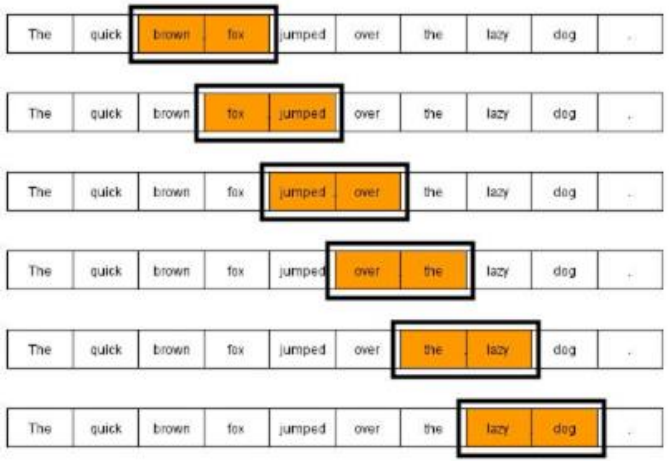
- 문장(Sentence)를 기준으로 토큰화
- 온점(.), 느낌표(!), 물음표(?) 등으로 분류하면 해결 될 것으로 생각됨
- 하지만 단순하게 분리할 경우 정확한 분리가 어려움

n-Gram

문서의 표현(Document Representation)

n-Gram 이란?

- BoW와 TDM은 개별 단어 하나만 바라보는 방법임
- 복수개(n개) 단어를 보는냐에 따라 unigram, bigram, trigram 등 으로 구분



n-Gram 이란?

- unigrams : an, adorable, little, boy, is, spreading, smiles
- bigrams : an adorable, adorable little, little boy, boy is, is spreading, spreading smiles
- trigrams : an adorable little, adorable little boy, little boy is, boy is spreading, is spreading smiles
- 4-grams : an adorable little boy, adorable little boy is, little boy is spreading, boy is spreading smiles

n-Gram 한계

- n의 크기는 trade-off 문제
  - 1보다는 2를 선택하는 것이 대부분 언어 모델 성능을 높일 수 있음
  - n을 너무 크게 선택하면 n-gram 이 unique 할 확률이 높아 등장수가 낮을 확률이 높음.
    (OOV, Out of Vocabulary 문제가 발생할 수 있음)
  - n을 너무 작게 하면 카운트는 잘되지만 정확도가 떨어질 수 있음. n은 최대 5를 넘지 않도록 권장

-	Unigram	Bigram	Trigram
Perplexity	962	170	109

스탠포드에 3,800만개 단어 토큰을 n-Gram으로 학습한 결과

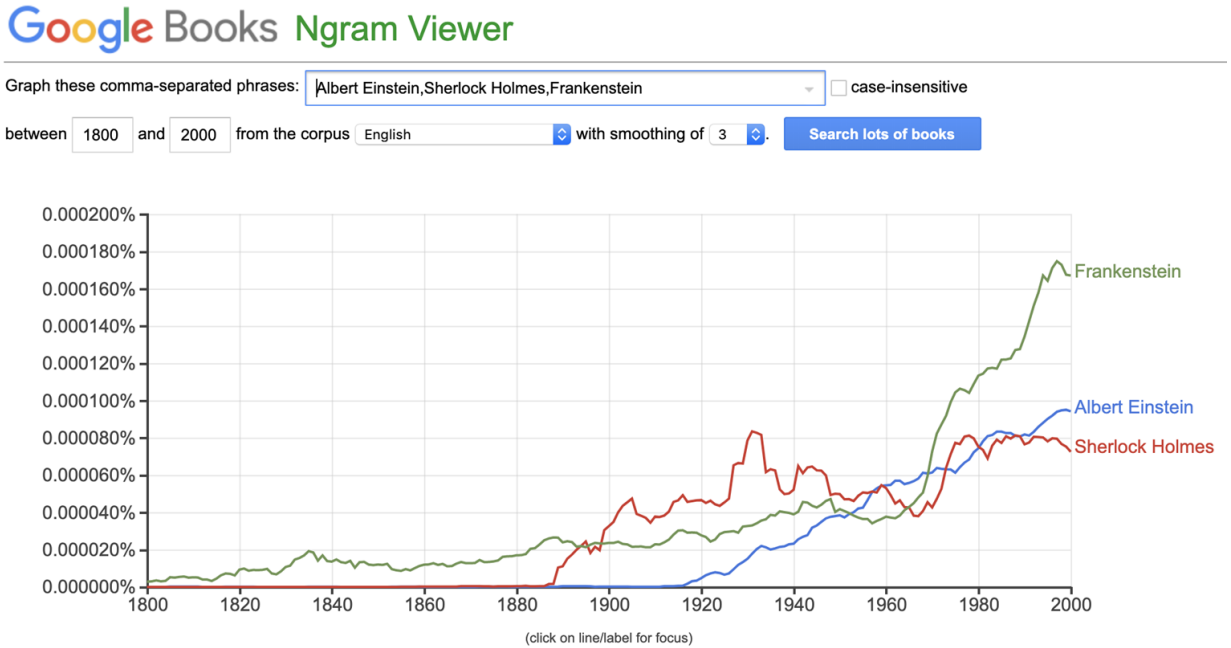
- n-Gram 카운트가 0인 경우
  - n-Gram 이 모든 단어를 커버 할 수 없기 때문에 Out of Vocabulary 문제가 발생할수 있음

적용 분야(Domain)에 맞는 코퍼스의 수집

- 분야(Domain)에 따라 단어들의 확률 분포는 다름
  (금융 분야는 금융 관련 용어가 많이 등장하고, 마케팅은 관련 용어가 많이 등장할 것임)

- 분야에 적합한 코퍼스를 사용하면 언어 모델의 성능이 높아질 수 있음
  (훈련에 사용되는 코퍼스에 따라 언어 모델의 성능이 달라짐 이는 언어 모델의 약점으로 분류되기도함)

Google Books Ngram Viewer



https://books.google.com

품사태깅
(PoS Tagging)

텍스트 전처리 (Text Preprocessing)

품사 부착(PoS Tagging)

- 각 토큰에 품사 정보를 추가
- 분석시에 불필요한 품사를 제거하거나 (예. 조사, 접속사 등) 필요한 품사를 필터링 하기 위해 사용

Barack Obama likes fried chicken very much.

Barack / NNP	Obama / NNP	likes / VBZ	fried / VBN	chicken / JJ	very / RB	much / RB	.

개체명 인식
(NER, Named Entity Recognition)

텍스트 전처리 (Text Preprocessing)

개체명 인식 (NER, Named Entity Recognition)

- 사람, 조직, 지역, 날짜, 숫자 등 개체 유형을 식별
- 검색 엔진 색인에 활용

Barack Obama likes fried chicken very much.

Barack / NNP / PERSON	Obama / NNP / ORGANIZATION	likes / VBZ	fried / VBN	chicken / JJ	very / RB	much / RB	.

원형 복원
(Stemming & Lemmatization)

텍스트 전처리 (Text Preprocessing)

어간 추출 (Stemming)

- 각 토큰의 원형 복원을 함으로써 토큰을 표준화하여 불필요한 데이터 중복을 방지
  (=단어의 수를 줄일수 있어 연산을 효율성을 높임)
- 어간 추출(Stemming) : 품사를 무시하고 **규칙에 기반하여 어간을 추출**
  규칙 : https://tartarus.org/martin/PorterStemmer/def.txt

원문	Stemming
running	run
beautiful	beauti
believes	believ
using	use
conversation	convers
organization	organ
studies	studi

표제어 추출 (Lemmatization)

- 각 토큰의 원형 복원을 함으로써 토큰을 표준화하여 불필요한 데이터 중복을 방지
  (=단어의 수를 줄일수 있어 연산을 효율성을 높임)
- 표제어 추출 (Lemmatization) : **품사정보를 유지하여 표제어 추출 (사전 기반)**

원문	Lemmatization
running	running
beautiful	beautiful
believes	belief
using	using
conversation	conversation
organization	organization
studies	study

불용어 처리
(Stopwords)

텍스트 전처리 (Text Preprocessing)

불용어 처리(Stopwords)

- 불필요한 토큰을 제거 하는 작업
- 불필요한 품사를 제거 하기도 함