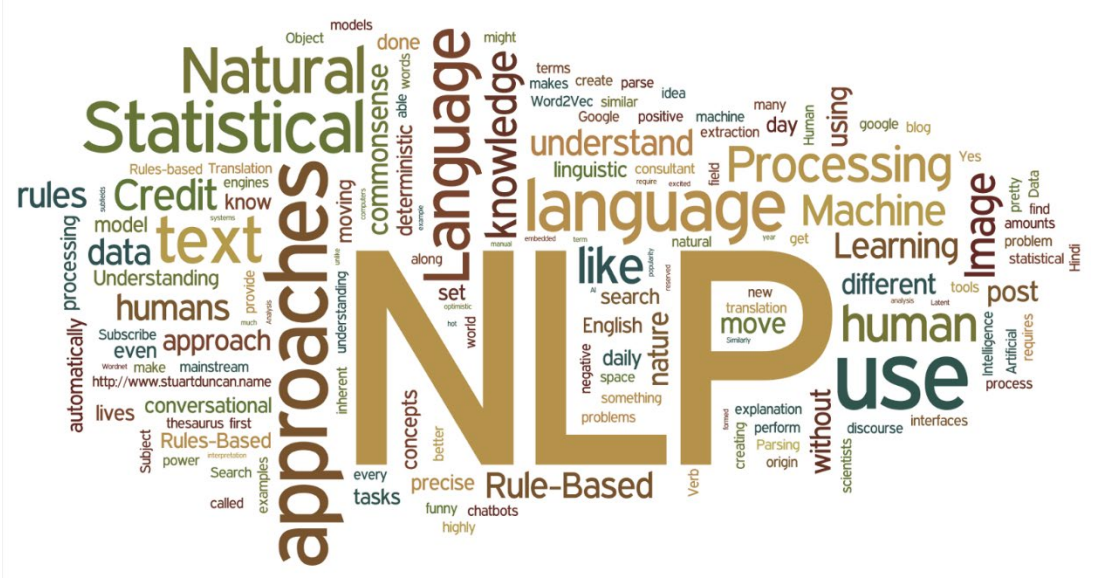
Data Science Lab
텍스트 마이닝



질문은 카톡방을 이용해주세요!

2020. 01. 30. (목)
발표자 : 이관용

TABLE OF CONTENTS

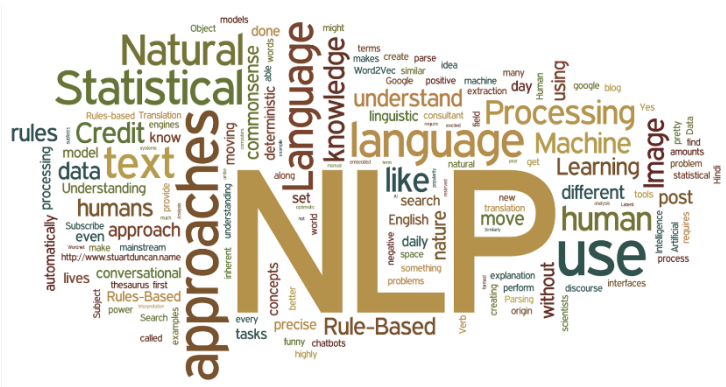
01
CONTENT

텍스트 분석 소개

기계학습을 사용하지 않는 방법

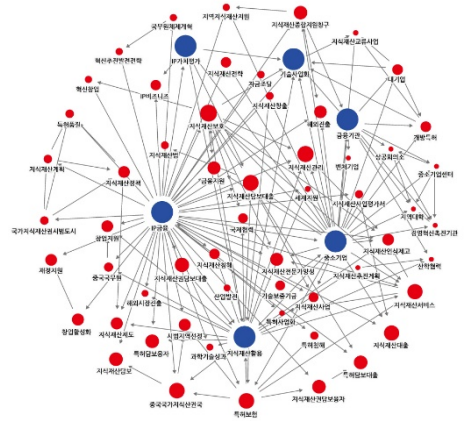
빈도 분석
(Keyword Frequency Analysis)

- 텍스트 전처리 결과를 통해 나온 **단어의 출현 횟수**를 분석
- 문서가 대략적으로 어떤 **주제**를 다루는지 알아보기 위함
- **워드클라우드**를 통한 시각화 가능



네트워크 분석
(Network Analysis)

- 텍스트에만 적용되는 방법은 아님
- 텍스트 분석에 사용될 경우 **단어 간 혹은 문서 간 관계**를 파악할 수 있음



01
CONTENT

텍스트 분석 소개

기계학습을 사용하는 방법

- 기계학습을 사용하는 방법을 이용하기 위해선 먼저 문서 혹은 단어를 벡터화 하는 'Vectorization' 과정이 필요
- Vectorization은 총 3가지 방법(Binary, TF, TF*IDF)이 존재
- Binary 방법은 해당 문서(corpus)에서 특정 단어가 존재하는 유무만 0, 1로 표현 → 잘 사용되지 않음

doc1 : 'apple banana orange carrot'
doc2 : 'orange orange banana carrot'
doc3 : 'tomato apple potato tomato'

TF
(Term Frequency)

- 문서에 나온 단어들의 **절대적 빈도**를 이용해 벡터로 만듦
- 위의 doc1 ~ doc3를 TF 벡터로 만들면 다음과 같음

	apple	banana	carrot	orange	potato	tomato
doc1	1	1	1	1	0	0
doc2	0	1	1	2	0	0
doc3	1	0	0	0	1	2

TF*IDF
(Term Frequency * Inverse Document Frequency)

- 해당 문서에서 특정 단어가 다른 문서에 비해 **상대적으로 얼마나 많이 나타났는지**를 표현
- **TF * 1/(DF+1)**로 나타낼 수 있음
  DF는 해당 문서 제외 다른 문서에서 해당 단어가 사용된 횟수

	apple	banana	carrot	orange	potato	tomato
doc1	1 * (1/2)	1 * (1/2)	1 * (1/2)	1 * (1/3)	0	0
doc2	0	1 * (1/2)	1 * (1/2)	2 * (1/2)	0	0
doc3	1 * (1/2)	0	0	0	1 * (1/1)	2 * (1/1)

01
CONTENT

텍스트 분석 소개
기계학습을 사용하는 방법

감성분석 (Sentiment Analysis)

- **문서의 긍/부정을 판별**하는 분석 방법
- 알고리즘을 사용하는 방법과 감성어 사전을 사용하는 방법
- **알고리즘 사용** : 미리 긍/부정을 나타내는 종속변수와 그에 맞는 텍스트를 준비한 후 분석
- **감성어 사전** : 신뢰도와 타당도가 높은 종속변수를 만들 수 없을 경우 감성어 사전 이용(한글 감성어 사전은 아직 제대로 개발되지 않음)

군집분석 (Clustering)

- 흔히 사용되는 **Kmeans Clustering을 텍스트에 적용**한 방법
- 앞서 언급한 Vectorization을 통해 문서별 벡터 생성
- **벡터 사이의 유사도를 기반으로 군집 분석**
- 유사도는 유클리디안 거리와 코사인 유사도를 기반으로 측정하지만 Scikit-learn에서는 유클리디안이 기본값

토픽 모델링 (Topic Modeling)

- 각 문서에서 어떠한 단어들이 어떤 빈도로 사용됐는지의 정보를 이용해 **문서의 주제**를 찾아내는 방법
- 컴퓨터가 주제를 알려주는 것이 아니라 '**주제별 단어 확률 분포**'와 '**문서별 주제 확률 분포**'에 대한 결과를 통해 분석하는 사람이 직접 주제를 찾아내야 함
- 조건부 확률을 기반으로 계산하기 때문에 **베이지안 통계**를 잘 알고 있어야 구체적 이해가 가능

워드 임베딩(Word Embedding)

- 단어를 벡터로 바꿔주는 것
- 방법은 크게 One-hot encoding과 **Distributed representation**이 있지만, One-hot encodin은 비효율적이라 이용하지 않음
- **Word2Vec**이 D/R 방법 중 하나인데, **앞 뒤에 사용된 단어가 같으면 문맥적 의미 역시 유사**하다고 보는 방법임
- 구글에서 **신경망 알고리즘**을 이용해 만든 방법론으로서, 페이스북에서 만든 FastText와 함께 자주 사용됨

02
CONTENT

텍스트 전처리

한글과 영어의 차이

교착어와 굴절어 외에 고립어(중국어), 포합어(뉴질랜드 원주민어)가 존재

한글

- **교착어** : 새로운 기능을 하는 단어를 표현하기 위해 형태소를 합침
- 예) 나는 → 나의, 나에게

영어

- **굴절어** : 기존 단어를 이용해 새로운 단어를 표현할 때 단어의 형태를 변화시킴
- 예) I → my, me, mine

- 위와 같은 차이는 **텍스트 전처리 단계의 차이를 발생**시킴
- **영어 : 문장 > 단어**
- **한글 : 문장 > 어절 > 단어 > 형태소**
- 명사(체언)는 그 자체가 형태소이지만, 형용사와 동사(용언)는 어간+어미로 나뉘어 어간이 분석의 단위인 형태소로 이용됨
- 텍스트 분석 효율 : 한글(60~70%) < 영어(90% 이상)
- **텍스트 전처리 결과물에서 한글과 영어의 차이는 없음**

02
CONTENT

텍스트 전처리

영어 텍스트 단계별 처리

nltk 모듈 이용

① Text Cleaning : 불필요한 기호나 심볼 제거 (!, @, “, .., ~, ^, (등)

② Case Conversion : 대소문자 통일 (영어에만 해당하는 부분)

③ Tokenization

: 형태소 분석 단계로서, 영어에서는 빈칸(띄어쓰기)을 이용해 문서를 token 단위로 쪼개는 작업

④ POS(Parts of Speech) tagging

: 품사를 태깅하는 작업으로서, 명사는 ‘NN’과 같은 값을 각 단어별로 태깅하고, 분석 목적에 맞는 품사를 뽑아내는 작업

⑤ Lemmatization

: 단어의 원형을 찾는 단계로서, 영어는 3인칭 단수형이 기본형

⑥ Removing Stopwords

: 불용어(不用語)를 제거하는 작업으로서, 분석에 별 의미가 없는 단어를 제거
예) 뉴스 기사 분석 시 신문사 이름, 기자 이름, 기자 이메일 주소의 도메인

02
CONTENT

텍스트 전처리

한글 텍스트 단계별 처리

Konlpy 모듈 이용

- 기본적으로 한글의 전처리 단계는 영어와 동일
- 대소문자를 일치시키는 Case Conversion 단계는 한글에 없음
- 한글 텍스트 분석 모듈인 **Konlpy는 Tokenization, POS tagging, Lemmatization을 형태소 분석기가 한 번에 처리해 간편함**
- **형태소 분석기** : 형태소 정보, 품사 정보, 생성/변형 정보 등을 담고 있는 사전
  Konlpy에는 총 5개 형태소 분석기 존재 (Komoran과 Twitter를 주로 이용)
- http://konlpy.org/en/latest/morph/#pos-tagging-with-konlpy
- **Komoran과 Twitter**(현재 Okt)는 속도가 느린데도 불구하고 미등록 단어를 추가 사전으로 등록해 이용할 수 있는 장점이 존재
- **미등록 단어(OOV, Out of Vocabulary) 문제** : 한글 텍스트 분석에서 크리티컬 사전에 **특정 형태소가 등록되지 않아 발생**하며, 주로 신조어, 고유명사, 전문 용어가 사전에 없을 때가 많은데, 이를 해결하는 것이 한글 텍스트 분석의 핵심

이상으로
이론 발표를 마칩니다.

감사합니다.