

# 1. 개요

---

## ❖ NLP(Natural Language Processing)

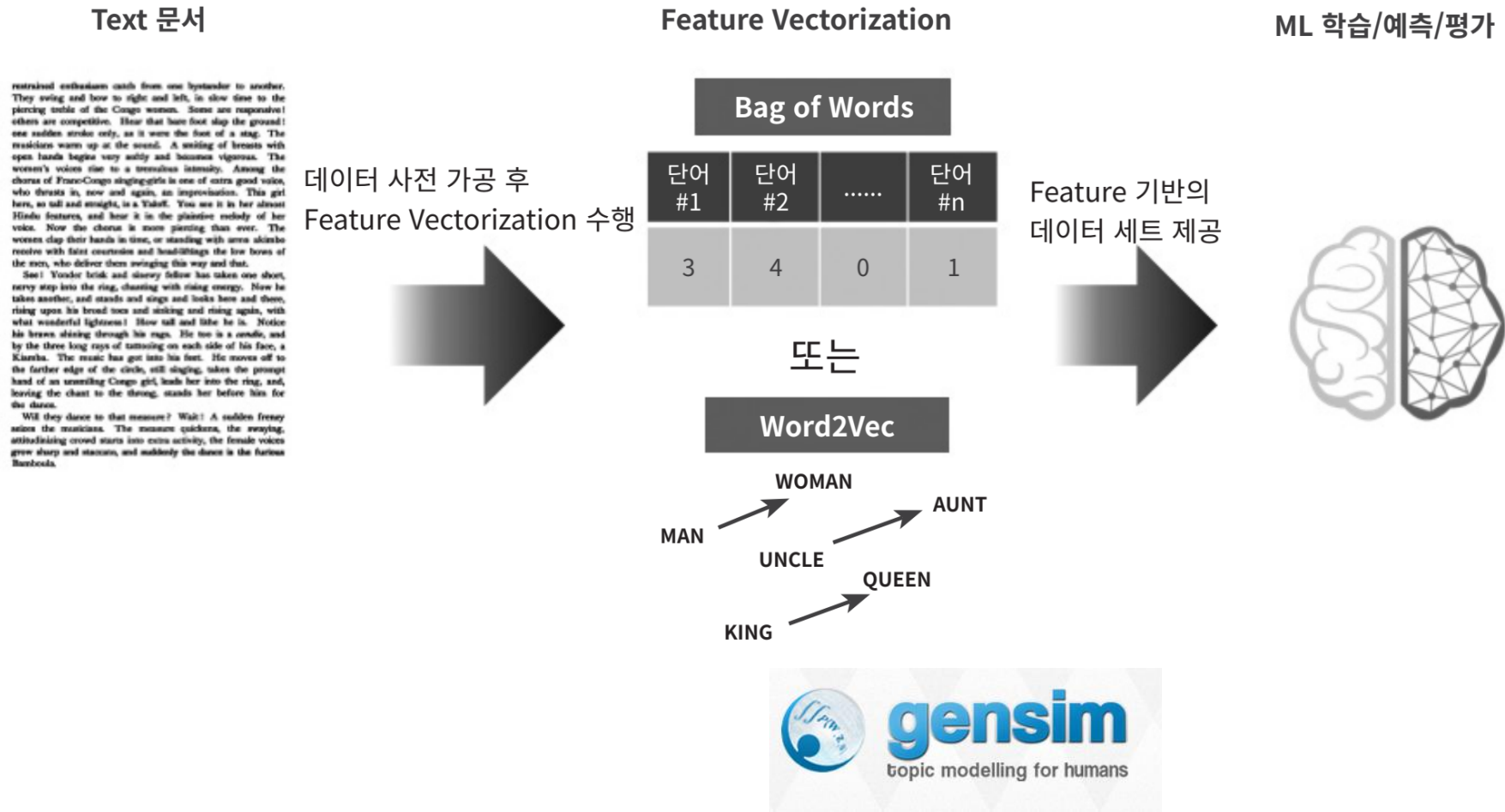
- 머신이 인간의 언어를 이해하고 해석하는 데 중점을 두고 발전
- 기계 번역, 질의 응답 시스템
- 텍스트 분석을 향상시켜주는 기반 기술

## ❖ 텍스트 분석(Text Analysis)

- 비정형 텍스트에서 의미있는 정보를 추출하는 것에 중점을 두고 발전
- 비즈니스 인텔리전스(BI)나 예측 분석 등의 분석 작업을 수행
- 텍스트 분류
- 감성 분석
- 텍스트 요약
- 텍스트 군집화와 유사도 측정

## 2. 텍스트 분석

### ❖ 텍스트 분석 프로세스



## 2. 텍스트 분석

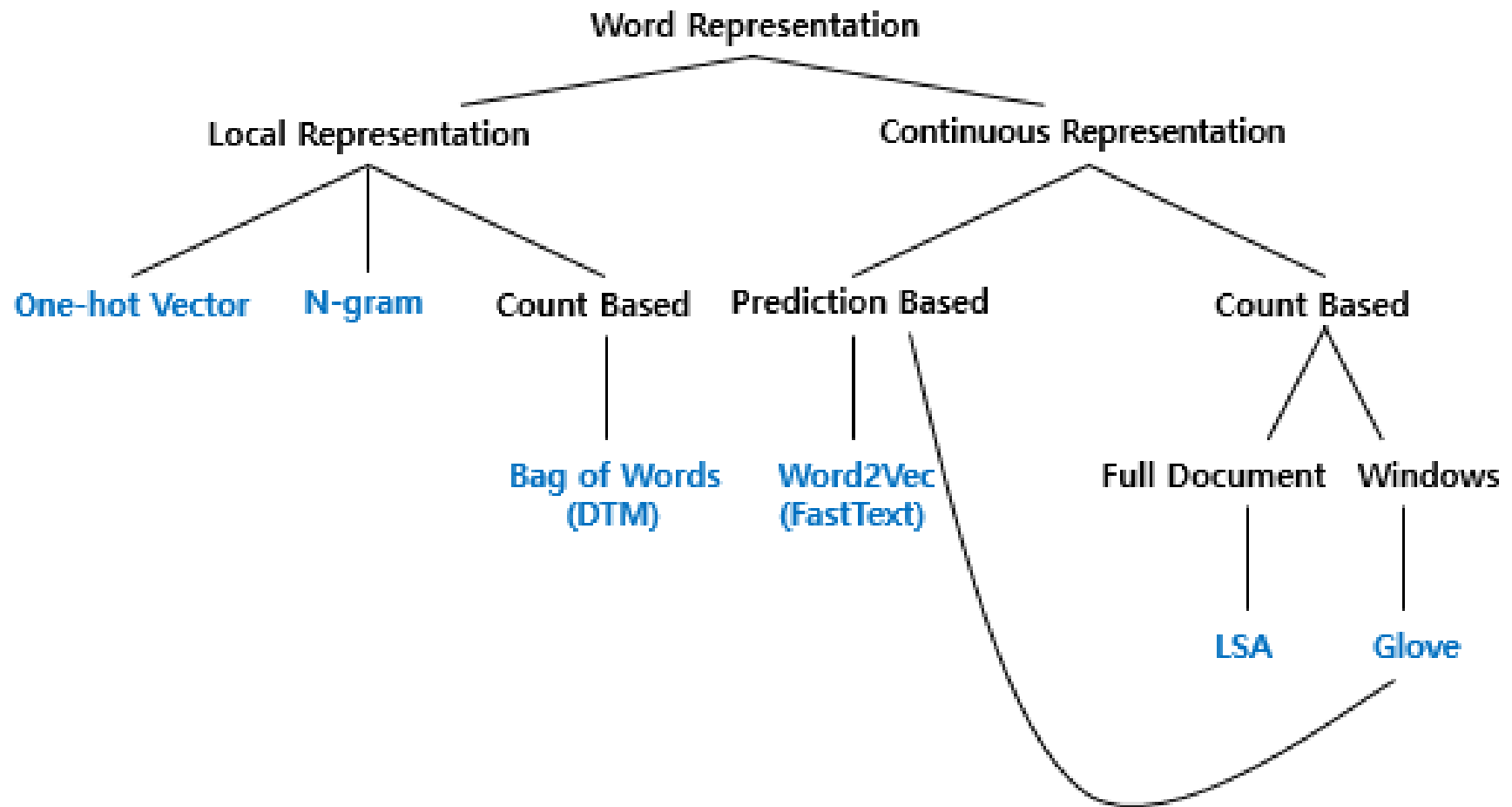
---

### ❖ 텍스트 전처리

- 클렌징(Cleansing)
- 토큰화(Tokenization)
  - 문장 토큰화
  - 단어 토큰화
- 필터링, 스톱 워드(불용어) 제거, 철자 수정
- Stemming
- Lemmatization

### 3. Bag of Words - BOW

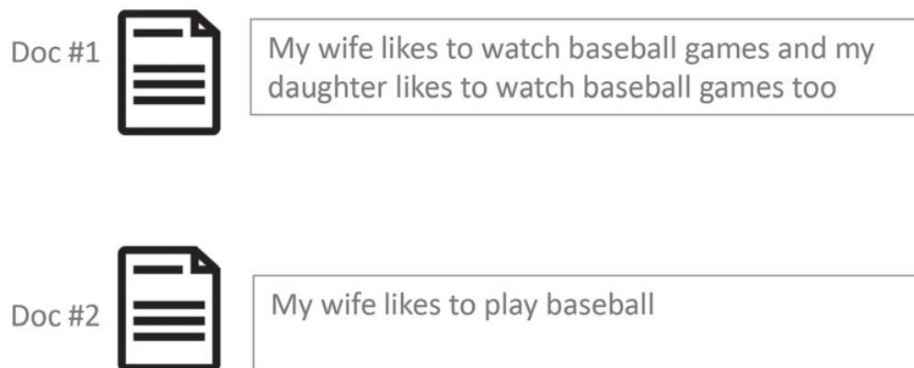
#### ❖ 단어의 표현 방법



### 3. Bag of Words - BOW

- 모든 단어를 문맥이나 순서를 무시하고 일괄적으로 단어에 대해 빈도 값을 부여해 피쳐 값을 추출하는 모델

여러 Document(문서)들과 Text들



	Index 0	Index 1	Index 2	Index 3	Index 4	Index 5	Index 6	Index 7	Index 8	Index 9	Index 10
	and	baseball	daughter	games	likes	my	play	to	too	watch	wife
문장 1	1	2	1	2	2	2		2	1	2	1
문장 2		1			1	1	1	1			1

→ 문장 1에서 baseball은 2회 나타남

### 3. Bag of Words - BOW

---

#### ❖ 장단점

##### ▪ 장점

- 쉽고 빠른 구축
- 의외로 높은 효율

##### ▪ 단점

- 문맥 의미(Semantic Context) 반영 부족
- 문장의 순서가 무시됨
  - I work at google.
  - I google at work.
- 오타, 줄임말에 취약함
- 희소 행렬 문제

### 3. Bag of Words - BOW

#### ❖ 피쳐 벡터화



- 카운트 기반의 벡터화
- TF-IDF(Term Frequency – Inverse Document Frequency) 기반의 벡터화

### 3. Bag of Words - BOW

#### ❖ Scikit-Learn CountVectorizer class

파라미터 명	파라미터 설명
max_df	<p><u>전체 문서에 걸쳐서 너무 높은 빈도수를 가지는 단어 피처를 제외하기 위한 파라미터입니다.</u> 너무 높은 빈도수를 가지는 단어는 스톱 워드와 비슷한 문법적인 특성으로 반복적인 단어일 가능성이 높기에 이를 제거하기 위해 사용됩니다.</p> <p>max_df = 100과 같이 정수 값을 가지면 전체 문서에 걸쳐 100개 이하로 나타나는 단어만 피처로 추출합니다. Max_df = 0.95와 같이 부동소수점 값(0.0 ~ 1.0)을 가지면 전체 문서에 걸쳐 빈도수 0~95%까지의 단어만 피처로 추출하고 나머지 상위 5%는 피처로 추출하지 않습니다.</p>
min_df	<p><u>전체 문서에 걸쳐서 너무 낮은 빈도수를 가지는 단어 피처를 제외하기 위한 파라미터입니다.</u> 수백~수천 개의 전체 문서에서 특정 단어가 min_df에 설정된 값보다 적은 빈도수를 가진다면 이 단어는 크게 중요하지 않거나 가비지(garbage)성 단어일 확률이 높습니다.</p> <p>min_df = 2와 같이 정수 값을 가지면 전체 문서에 걸쳐서 2번 이하로 나타나는 단어는 피처로 추출하지 않습니다. min_df = 0.02와 같이 부동소수점 값(0.0 ~ 1.0)을 가지면 전체 문서에 걸쳐서 하위 2% 이하의 빈도수를 가지는 단어는 피처로 추출하지 않습니다.</p>
max_features	<p>추출하는 피처의 개수를 제한하며 정수로 값을 지정합니다. 가령 max_features = 2000으로 지정할 경우 가장 높은 빈도를 가지는 단어 순으로 정렬해 2000개까지만 피처로 추출합니다.</p>
stop_words	<p>'english'로 지정하면 영어의 스톱 워드로 지정된 단어는 추출에서 제외합니다.</p>



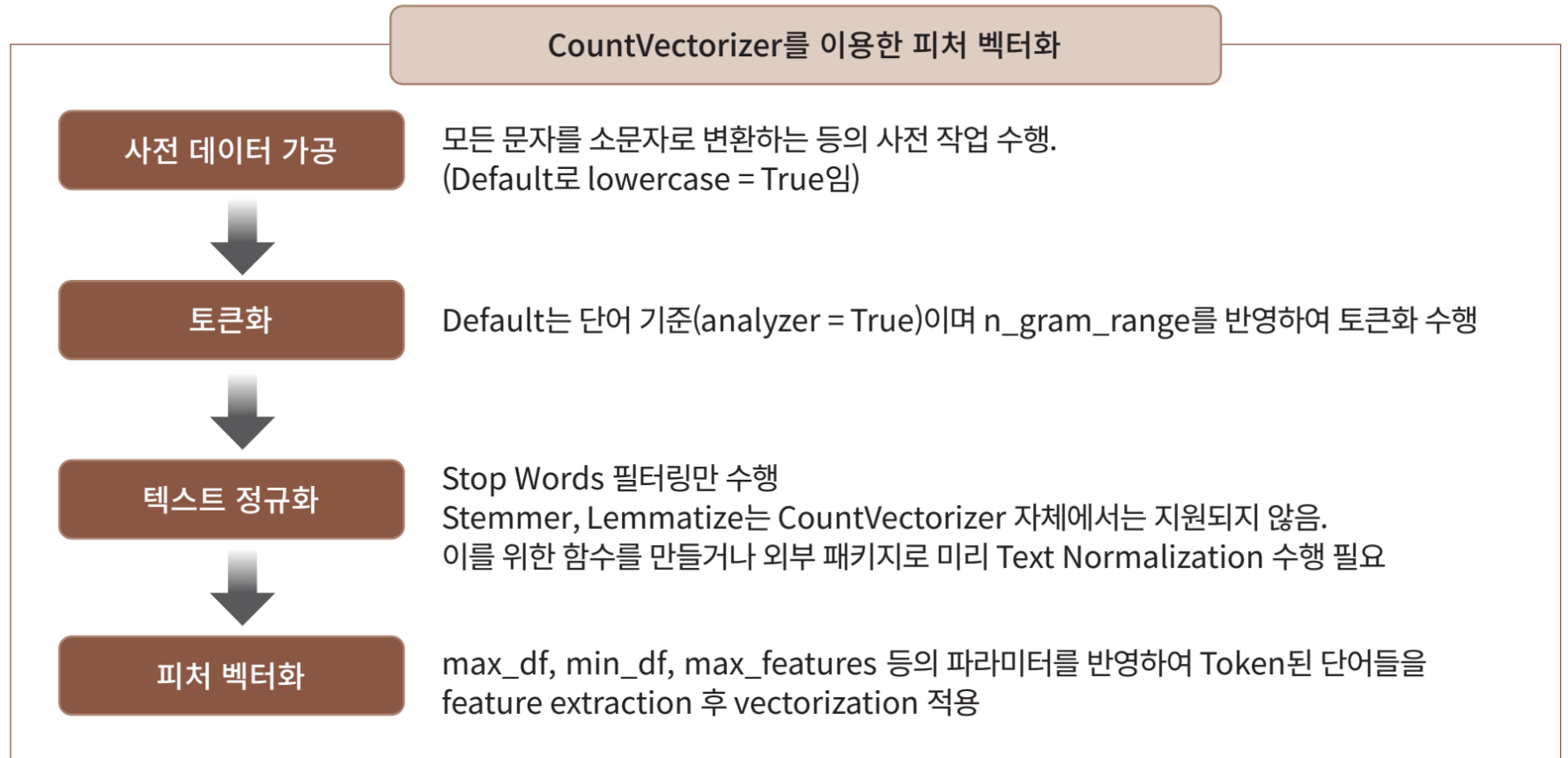
### 3. Bag of Words - BOW

#### ❖ Scikit-Learn CountVectorizer class

n_gram_range	<p><u>Bag of Words 모델의 단어 순서를 어느 정도 보강하기 위한 n_gram 범위를 설정합니다. 튜플 형태로 (범위 최솟값, 범위 최댓값)을 지정합니다.</u></p> <p>예를 들어 (1, 1)로 지정하면 토큰화된 단어를 1개씩 피처로 추출합니다. (1, 2)로 지정하면 토큰화된 단어를 1개씩(minimum 1), 그리고 순서대로 2개씩(maximum 2) 묶어서 피처로 추출합니다.</p>
analyzer	<p>피처 추출을 수행한 단위를 지정합니다. 당연히 디폴트는 'word'입니다. Word가 아니라 character의 특정 범위를 피처로 만드는 특정한 경우 등을 적용할 때 사용됩니다.</p>
token_pattern	<p>토큰화를 수행하는 정규 표현식 패턴을 지정합니다. 디폴트 값은 '\b\w\w+\b'로, 공백 또는 개행 문자 등으로 구분된 단어 분리자(\b) 사이의 2문자(문자 또는 숫자, 즉 영숫자) 이상의 단어(word)를 토큰으로 분리합니다. analyzer= 'word'로 설정했을 때만 변경 가능하나 디폴트 값을 변경할 경우는 거의 발생하지 않습니다.</p>
tokenizer	<p>토큰화를 별도의 커스텀 함수로 이용시 적용합니다. 일반적으로 CountTokenizer 클래스에서 어근 변환 시 이를 수행하는 별도의 함수를 tokenizer 파라미터에 적용하면 됩니다.</p>

### 3. Bag of Words - BOW

#### ❖ CountVectorizer를 이용한 피처 벡터화



## 4. n-그램

### ❖ n-그램

- 연속적인  $n$  개의 토큰으로 구성된 단어, 문자

**fine thank you**

- 1-gram (Unigram)
  - Word level: [fine, thank, you]
  - Character level: [f, i, n, e, , t, h, a, n, k, , y, o, u]
- 2-gram (Bigram)
  - Word level: [fine thank, thank you]
  - Character level: [fi, in, ne, e , t, th, ha, an, nk, k , y, yo, ou]
- 3-gram (Trigram)
  - Word level: [fine thank you]
  - Character level: [fin, ine, ne , e t, th, tha, han, ank, nk , k y, yo, you]

## 4. n-그램

---

### ❖ 사용 이유

- Bag of Words 의 약점 극복
- 다음 단어 예측
- 오타 발견
- 단어 추천

## 4. n-그램

### ❖ Bag of Words 약점

- machine learning is fun and is not boring

machine	fun	is	learning	and	not	boring	...
1	1	2	1	1	1	1	...

- machine learning 조합을 찾기 어려움
- not 이 어디에 위치하는지 모름
- "machine is boring and learning is not fun" 과 구분이 안됨

- Bag of Bigram

machine learning	learning is	is fun	fun and	and is	is not	not boring	...
1	1	1	1	1	1	1	...

## 4. n-그램

### ❖ Naïve next word prediction

- how are you doing
- how are you
- how are they

Trigram	횟수
how are you	2
are you doing	1
how are they	1

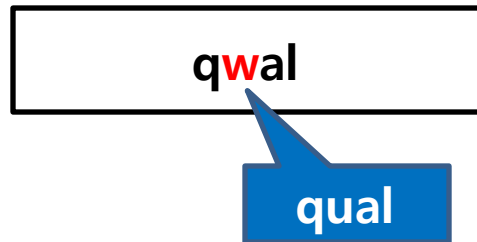
Input box

**how are you**

## 4. n-그램

### ❖ Naïve spell checker

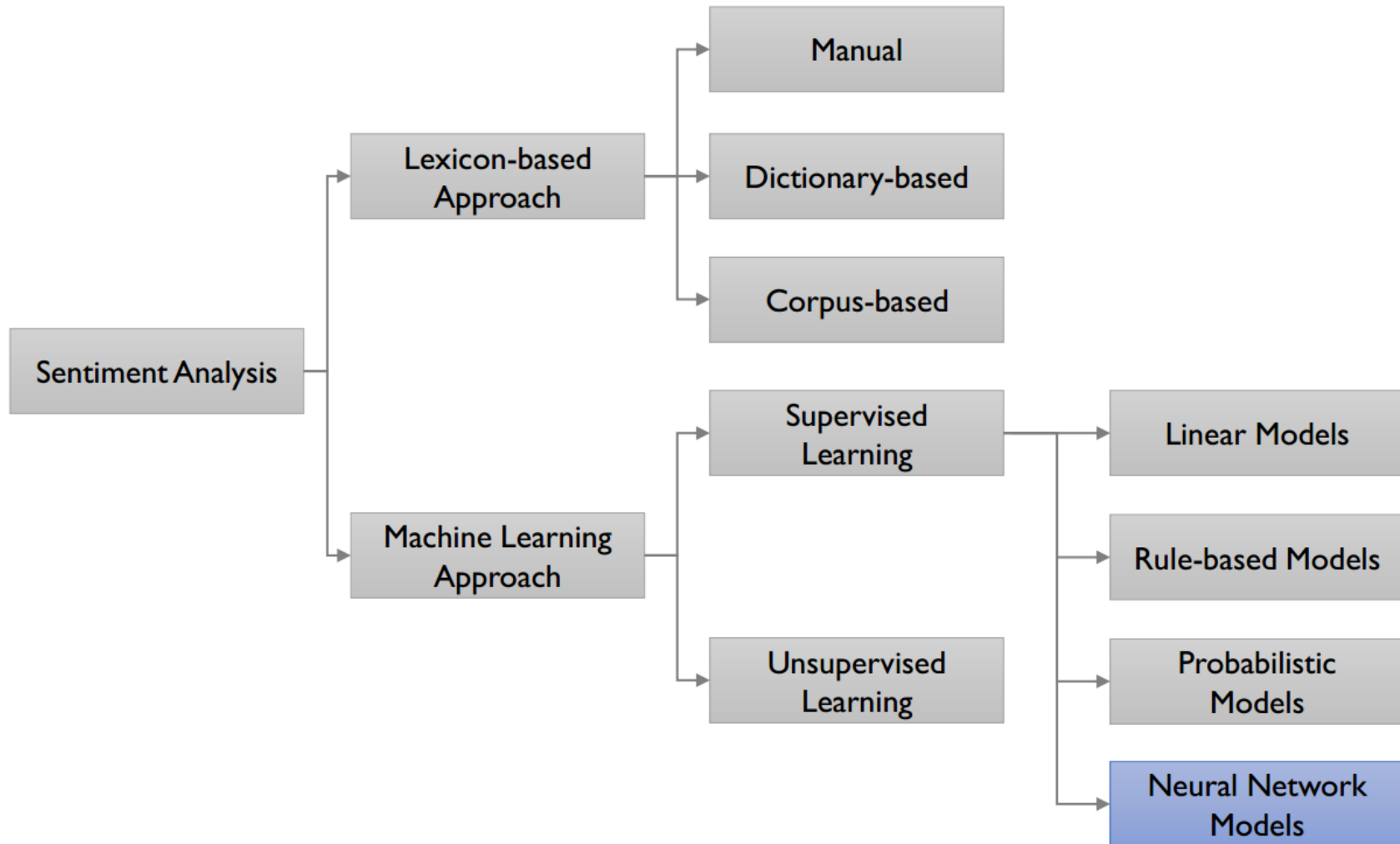
- quality
- quarter
- quit



Bigram	횟수
qu	3
ua	2
al	1
li	1
it	2
ty	1
ar	1
rt	1
te	1
er	1
ui	1

## 5. 감성 분석

### ❖ 종류





## 5. 감성 분석

### ❖ IMDB 영화평



## Bag of Words Meets Bags of Popcorn

Use Google's Word2Vec for movie reviews

578 teams · 3 years ago

Overview

Data

Kernels


Discussion

Leaderboard


Rules

## 6. 한글 감성 분석

### ❖ 네이버 영화 평점 감성 분석

 e9t Fix spacing typo in partition.py Latest commit cc0670e on Jun 28, 2016

code	Fix spacing typo in partition.py	2 years ago
raw	Add raw data	3 years ago
README.md	Upload README	3 years ago
ratings.txt	Initial commit	3 years ago
ratings_test.txt	Modify headers	3 years ago
ratings_train.txt	Modify headers	3 years ago
synopses.json	Add synopses data	3 years ago

 README.md

## Naver sentiment movie corpus v1.0

This is a movie review dataset in the Korean language. Reviews were scraped from Naver Movies.

The dataset construction is based on the method noted in Large movie review dataset from Maas et al., 2011.

### Data description

- Each file is consisted of three columns: `id`, `document`, `label`
  - `id`: The review id, provided by Naver
  - `document`: The actual review
  - `label`: The sentiment class of the review. (0: negative, 1: positive)
  - Columns are delimited with tabs (i.e., `.tsv` format; but the file extension is `.txt` for easy access for novices)

## 7. 텍스트 분류

### ❖ 20 뉴스그룹 분류



## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 조건부 확률

- B 사건이 발생했을 때 A 사건이 일어날 확률

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

- 또는, A 사건이 발생했을 때 B 사건이 일어날 확률

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

- A, B 사건이 독립인 경우에는 아래 식을 만족한다.

$$P(A \cap B) = P(A)P(B)$$

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 베이즈의 정리

#### ▪ 용어 정리

- 사전 확률 : 이미 알고 있는 사건(들)의 확률
- 우도(Likelihood Probability)

이미 알고 있는 사건(들)이 발생했다는 조건하에  
다른 사건이 발생할 확률

- 사후 확률 : 사전 확률과 우도 확률을 통해서 알게 되는 조건부 확률

#### ▪ 베이즈 정리(Bayes Theorem)

$$P(A_k|B) = \frac{P(B|A_k)P(A_k)}{P(B)}$$

주변우도 (Marginal Likelihood)

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 베이즈 예

	메일로부터 토큰화 및 정제된 단어	분류
1	me free lottery	spam
2	free get free you	spam
3	you free scholarship	ham
4	free to contact me	ham
5	you won award	ham
6	you ticket lottery	spam

- free가 들어간 메일이 스팸 메일일 확률을 구하시오.

$$P(\text{spam}|\text{free}) = (P(\text{free} | \text{spam}) * P(\text{spam})) / (P(\text{free})) = (2/3 * 3/6) / (4/6) = 1/2$$

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 나이브 베이즈

- 나이브(Naïve)의 의미
- '순진한', '순수한' 이라는 뜻
- 수학에서 단순성을 부여할 때 사용
- 다양한 세부 요인의 영향력을 모두 동등하고 독립적이라고 가정

### ❖ 나이브 베이즈 분류기의 예 - 스팸 메일 필터

- 입력 텍스트(메일의 본문)가 주어졌을 때

$P(\text{정상 메일} \mid \text{입력 텍스트}) = \text{입력 텍스트가 있을 때 정상 메일일 확률}$

$P(\text{스팸 메일} \mid \text{입력 텍스트}) = \text{입력 텍스트가 있을 때 스팸 메일일 확률}$

- 베이즈 정리에 따라 식을 표현하면

$P(\text{정상 메일} \mid \text{입력 텍스트}) = P(\text{입력 텍스트} \mid \text{정상 메일}) \times P(\text{정상 메일}) / P(\text{입력 텍스트})$

$P(\text{스팸 메일} \mid \text{입력 텍스트}) = P(\text{입력 텍스트} \mid \text{스팸 메일}) \times P(\text{스팸 메일}) / P(\text{입력 텍스트})$

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

- 입력 텍스트가 주어졌을 때,  $P(\text{정상 메일} \mid \text{입력 텍스트})$ 가  $P(\text{스팸 메일} \mid \text{입력 텍스트})$  보다 크면 정상 메일

$$P(\text{정상 메일} \mid \text{입력 텍스트}) = P(\text{입력 텍스트} \mid \text{정상 메일}) \times P(\text{정상 메일})$$

$$P(\text{스팸 메일} \mid \text{입력 텍스트}) = P(\text{입력 텍스트} \mid \text{스팸 메일}) \times P(\text{스팸 메일})$$

- 입력 텍스트는 메일의 본문

메일 본문에 있는 모든 단어를 토큰화시켜서 이 단어들을 나이브 베이즈 분류기의 입력으로 사용

- 만약 메일 본문에 있는 단어가 3개라고 가정( $w_1, w_2, w_3$ )

나이브 베이즈 분류기는 모든 단어가 독립적이라고 가정

$$P(\text{정상 메일} \mid \text{입력 텍스트}) = P(w_1 \mid \text{정상 메일}) \times P(w_2 \mid \text{정상 메일}) \times P(w_3 \mid \text{정상 메일}) \\ \times P(\text{정상 메일})$$

$$P(\text{스팸 메일} \mid \text{입력 텍스트}) = P(w_1 \mid \text{스팸 메일}) \times P(w_2 \mid \text{스팸 메일}) \times P(w_3 \mid \text{스팸 메일}) \\ \times P(\text{스팸 메일})$$



## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 스팸 메일 분류기

#### ▪ 훈련 데이터

	메일로부터 토큰화 및 정제된 단어	분류
1	me free lottery	spam
2	free get free you	spam
3	you free scholarship	ham
4	free to contact me	ham
5	you won award	ham
6	you ticket lottery	spam

#### ▪ 'you free lottery' 는 spam인가 ham인가?

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 스팸 메일 분류기

- 'you free lottery' 에 대해 정상 메일일 확률과 스팸 메일일 확률

$$P(\text{정상 메일} \mid \text{입력 텍스트}) = P(\text{you} \mid \text{정상 메일}) \times P(\text{free} \mid \text{정상 메일}) \\ \times P(\text{lottery} \mid \text{정상 메일}) \times P(\text{정상 메일})$$

$$P(\text{스팸 메일} \mid \text{입력 텍스트}) = P(\text{you} \mid \text{스팸 메일}) \times P(\text{free} \mid \text{스팸 메일}) \\ \times P(\text{lottery} \mid \text{스팸 메일}) \times P(\text{스팸 메일})$$

- $P(\text{정상 메일}) = P(\text{스팸 메일}) = \text{총 메일 6개 중 3개} = 0.5$
- $P(\text{you} \mid \text{정상 메일})$ 를 구하는 방법
  - 분모: 정상 메일에 등장한 모든 단어의 빈도 수의 총합
  - 분자: 정상 메일에서 you가 총 등장한 빈도 수

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 스팸 메일 분류기

- 정상 메일

you	free	scholar- ship	free	to	contact	me	you	won	award
2	2	1	-	1	1	1	-	1	1

- 스팸 메일

me	free	lottery	free	get	free	you	you	ticket	lottery
1	3	2	-	1	-	2	-	1	-

- “you free lottery”에 대한 확률 계산

$$P(\text{정상 메일} \mid \text{입력 텍스트}) = 2/10 \times 2/10 \times 0/10 = 0$$

$$P(\text{스팸 메일} \mid \text{입력 텍스트}) = 2/10 \times 3/10 \times 2/10 = 0.012$$

- $P(\text{정상 메일} \mid \text{입력 텍스트}) < P(\text{스팸 메일} \mid \text{입력 텍스트})$ 이므로 스팸 메일

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

---

### ❖ 장점

- 간단하고, 빠르며, 정확한 모델
- computation cost가 작음 (따라서 빠름)
- 큰 데이터셋에 적합
- 연속정보보다 이산형 데이터에서 성능이 좋음
- Multiple class 예측을 위해서도 사용 가능

### ❖ 단점

- feature 간의 독립성이 있어야 함

하지만 실제 데이터에서 모든 feature가 독립인 경우는 희박함

## 8. 나이브 베이즈 분류기(Naïve Bayes Classifier)

### ❖ 데이터 전처리

- 데이터를 토큰화한 후 BoW(Bag of Words)로 만들어 주어야 함

```
from sklearn.feature_extraction.text import CountVectorizer
cvector = CountVectorizer()
X_train_cvect = cvector.fit_transform(newsgroup.data)
```

### ❖ Scikit-Learn에서 제공하는 나이브 베이즈 모델

- 나이브 베이즈 분류 수행

```
from sklearn.naive_bayes import MultinomialNB # 다항분포 나이브 베이즈 모델
nb = MultinomialNB()
nb.fit(X_train_cvect, newsgroup.target)
```

- 정확도 측정

```
from sklearn.metrics import accuracy_score # 정확도 계산
pred = model.predict(X_test_cvect) # 테스트 데이터에 대한 예측
acc = accuracy_score(newsgroup_test.target, predicted)
print(f"정확도:{acc:.4f}" %) # 예측값과 실제값 비교
```