Text Analytics

Ch7 : Topic Modeling

서수원

Business Intelligence Lab.

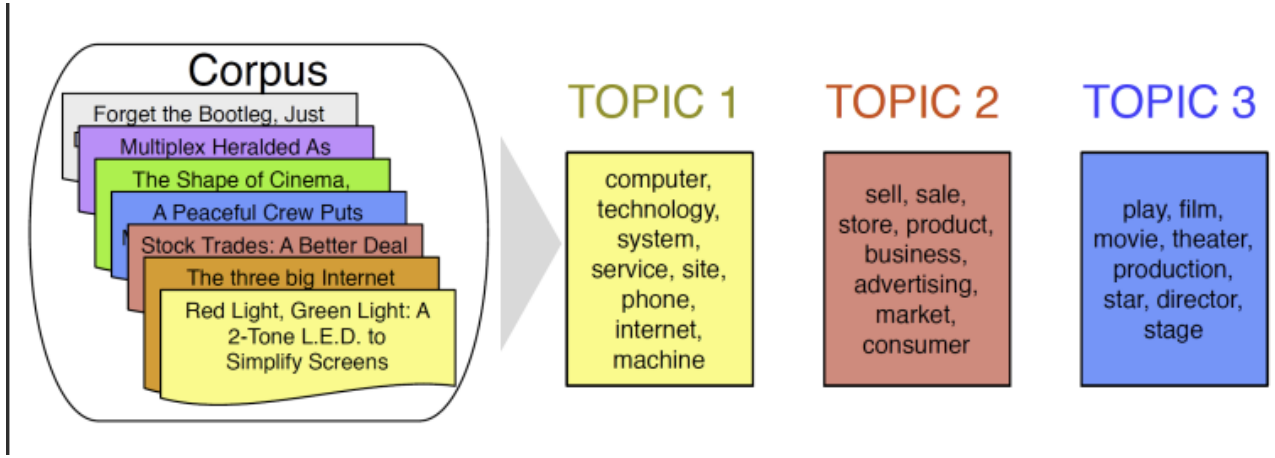
산업경영공학과, 명지대학교

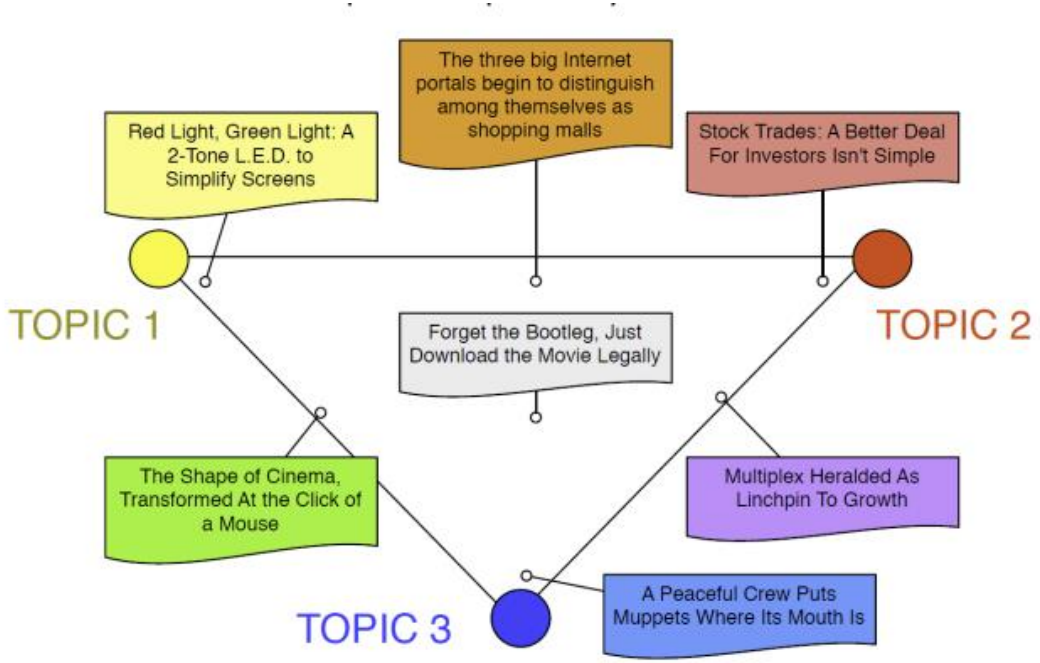
01

Topic Modeling

Topic Model

- Topic Model
  - 코퍼스에 대해 토픽의 수를 정의하는 것이 가능 하다면, 단어를 topic에 할당 가능하다.
  - 각 문서에 어떤 topic이 들어 있는지를 볼 수 있다.





Topic Model

- Topic Extraction
  - 딥러닝에 대한 논문 초록을 크롤링 해 토픽 모델링 한 예시이다.
    - ✓ 토픽의 제목은 사람이 직접 설정 해줘야 한다.

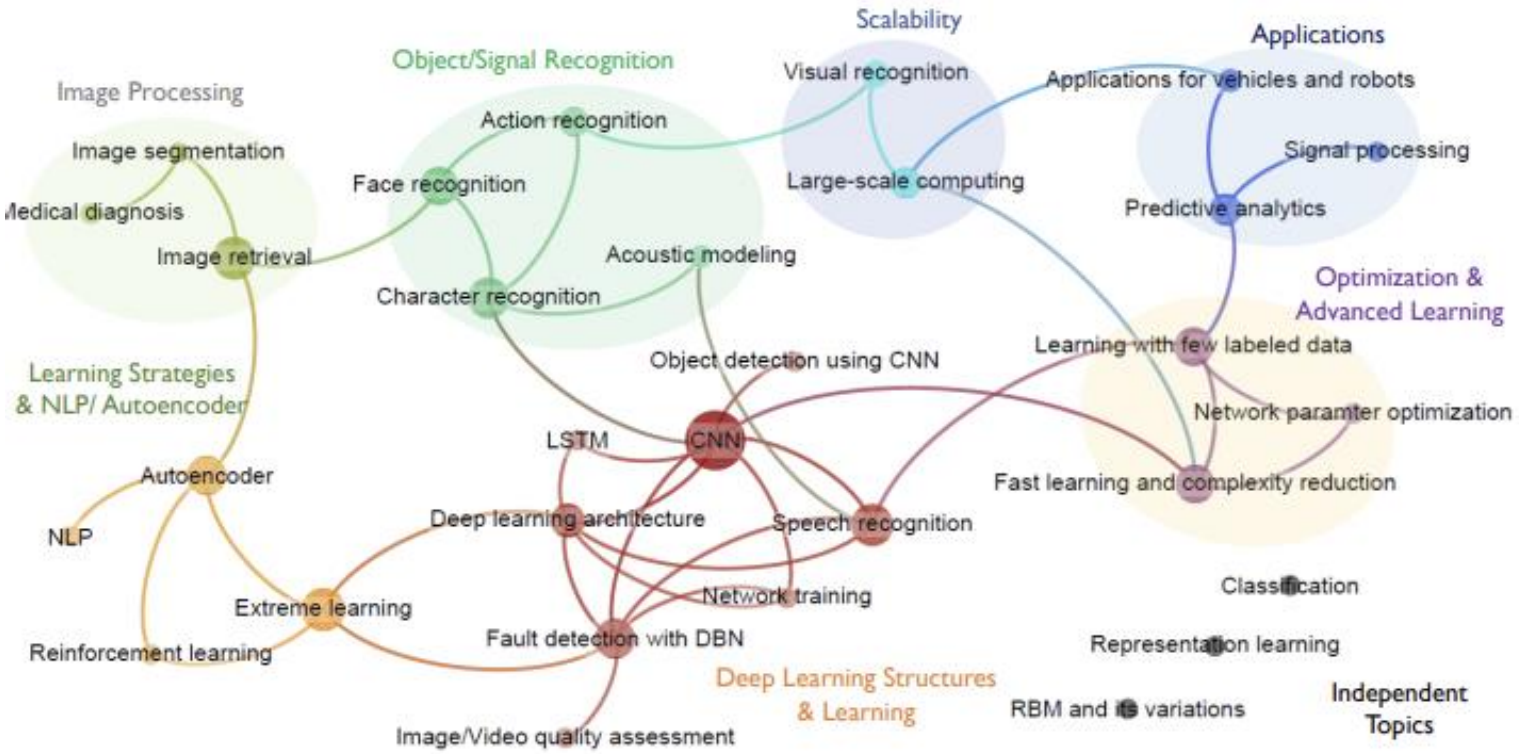
Fault detection with DBN	Convolutional neural network	Network Learning	Representation learning	Face Recognition	Speech Recognition	Acoustic Modeling	Extreme Learning	Deep learning architecture	Image Segmentation
deep	neural	layer	feature	face	speaker	speech	deep	deep	image
belief	convolutional	input	level	recognition	speech	recognition	learn	architecture	scene
network	pool	output	extract	estimation	noise	acoustic	algorithm	neural	scale
dbn	convolution	unit	learn	facial	adaptation	hmm	structure	standard	segmentation
fault	convnet	hide	extraction	shape	source	neural	extreme	explore	pixel

Long-short term memory	Predictive analytics	Signal processing	Classification models	Large-scale computing	Image quality assessment	Visual recognition	NLP	Detection using CNN	Action recognition
term	data	analysis	classification	application	domain	pattern	word	cnn	video
recurrent	prediction	filter	classifier	implementation	state	process	text	detection	human
long	technique	signal	class	efficient	quality	compute	language	convolutional	temporal
lstm	information	component	vector	process	resolution	visual	representation	neural	action
network	research	audio	support	power	relationship	field	semantic	detect	track

Image retrieval	Medical image diagnosis	Reinforcement learning	Parameter optimization	Auto encoder	RBM and variations	Learning with few labeled data	Fast learning complexity reduction	Applications for vehicles & robots	Character recognition
image	image	learn	train	representation	machine	train	fast	time	recognition
visual	segmentation	question	algorithm	learn	boltzmann	data	reduce	real	system
retrieval	disease	state	gradient	sparse	rbm	label	parameter	application	character
descriptor	cell	answer	sample	encode	restrict	few	weight	drive	network
attribute	medical	reinforcement	optimization	stack	distribution	transfer	complexity	Vehicle	neural

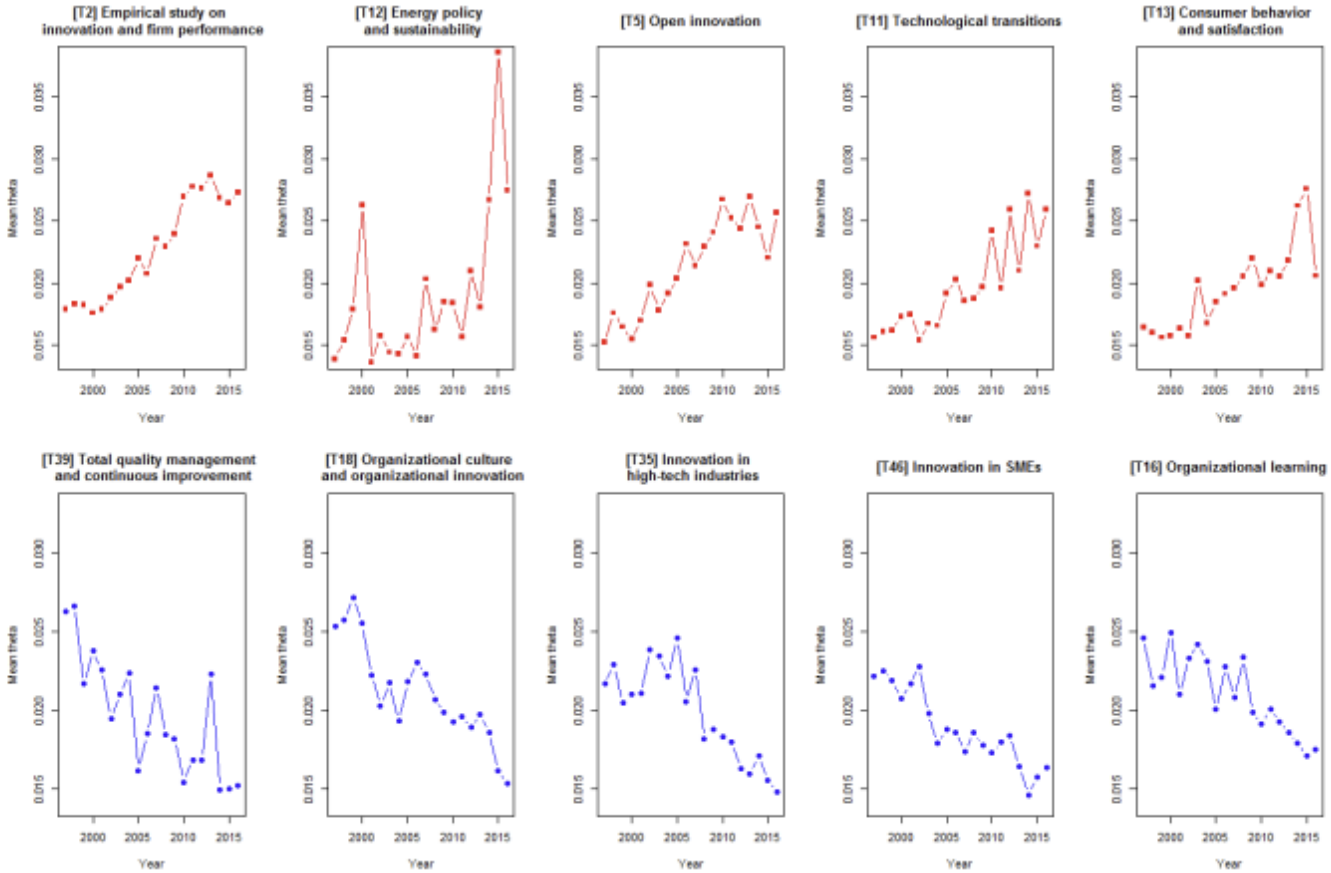
Topic Model

- Topic Relation between Topics
  - 여러 토픽들 사이의 관계(유사도)를 볼 수 있다.



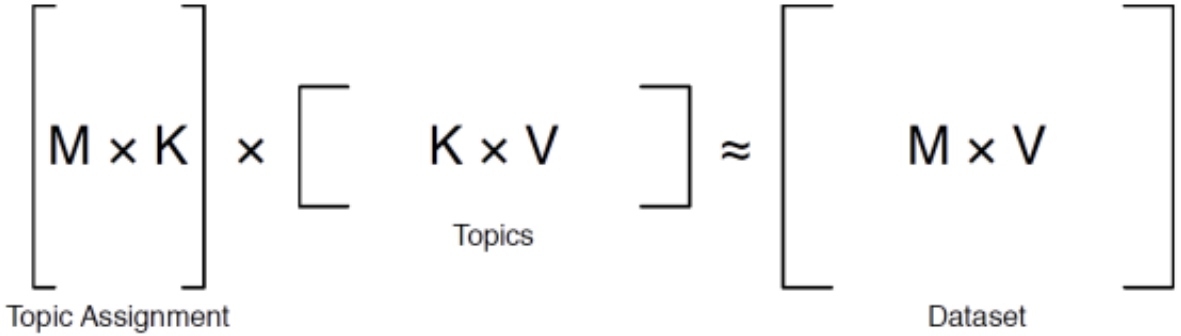
Topic Model

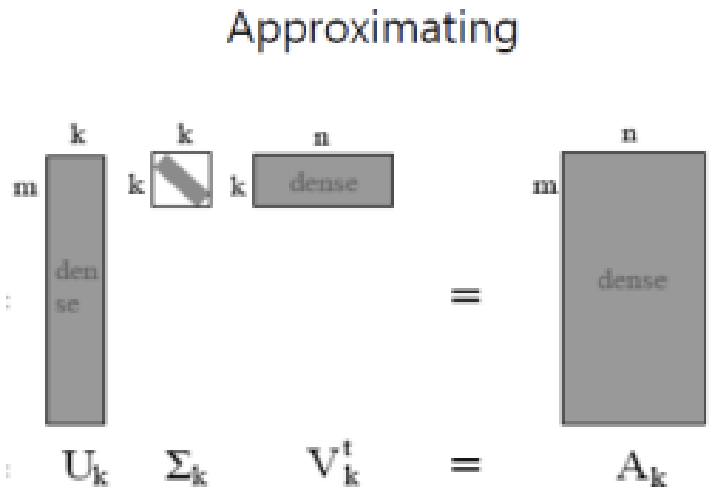
- Trend Analysis
  - 토픽의 트랜드를 볼 수 있다.
    - ✓ 예시는 '기술 혁신 경영' 이란 카테고리에서 토픽모델링을 한 결과이다.



Topic Model

- Matrix Factorization Approach
  - Topic Assignment 는 topics와 Dataset으로 표현된다.
- LSA
  - N개의 문서는 k개의 토픽으로 이루어져 있다.





Topic Model

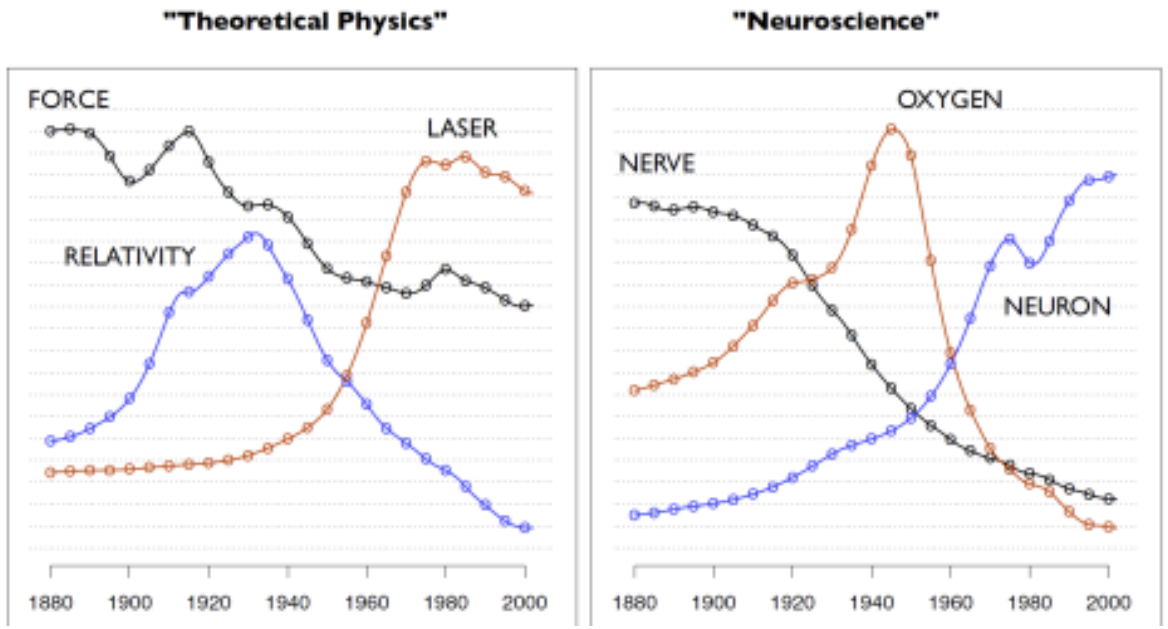
- LSA의 단점
  - 데이터가 정규분포여야 하는데 그렇지 않다.
    - ✓ Tf-IDF나 BOW를 사용하기 때문이다.
      - ❖ 희소성의 문제가 생긴다.
      - ❖ 단어의 등장 빈도가 정규분포가 아니다.
    - ✓ TF-IDF는 그럼에도 불구하고 신기하게 성능이 좋다.

Topic Model

- Probabilistic Topic Model
  - 각각의 문서는 토픽에 대한 확률분포이다.
  - 각각의 토픽은 단어에 대한 확률분포이다.

Topic "Education": school, students, education, university,…

Topic "Budget": million, finance, tax, program, …

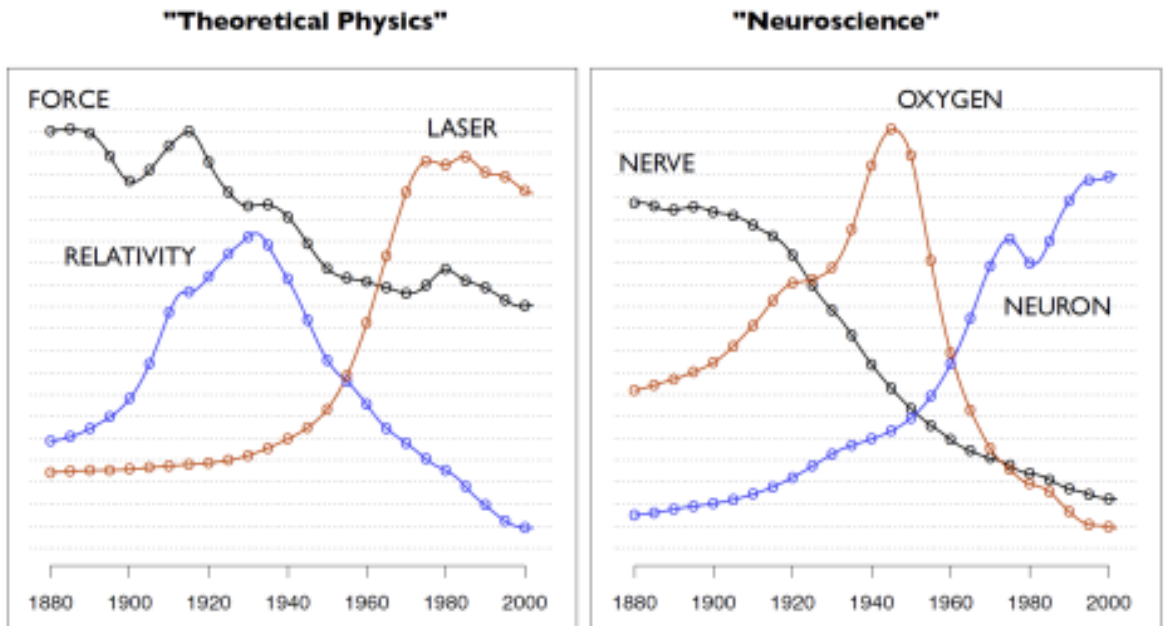


Topic Model

- Probabilistic Topic Model
  - 각각의 문서는 토픽에 대한 확률분포이다.
  - 각각의 토픽은 단어에 대한 확률분포이다.

Topic "Education": school, students, education, university,…

Topic "Budget": million, finance, tax, program, …

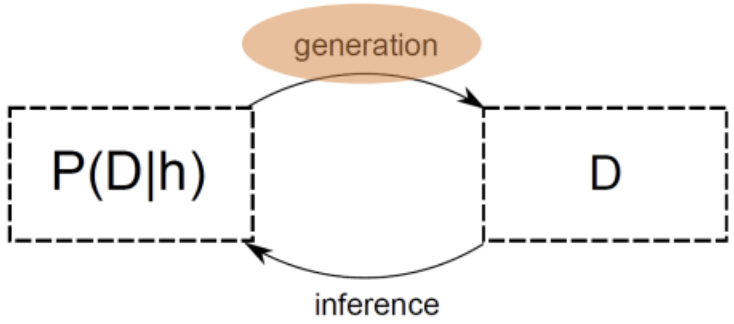


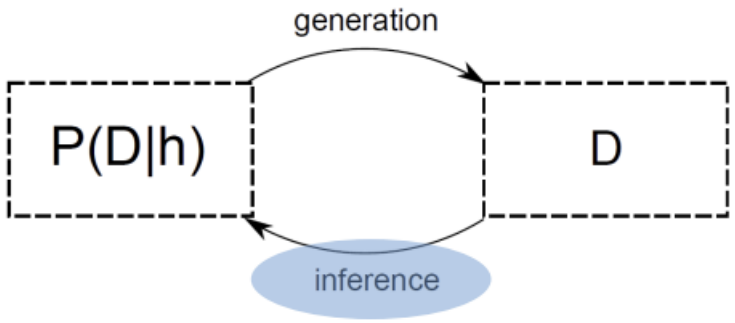
Topic Model

- Topic Model : Generative Approach
  - 데이터를 통해 우리가 가정한 확률적 모델에 잘 맞춰야 한다.
    - ✓ 확률값을 최대로 해야한다.
  - 생성모델은 조건부 확률을 정의한다.P(D|h)
    - ✓ H라는 가설로 D라는 데이터셋이 만들어진다.
    - ✓ H라는 가정을 추론 하는 것이 '학습'이 된다.

Topic Model

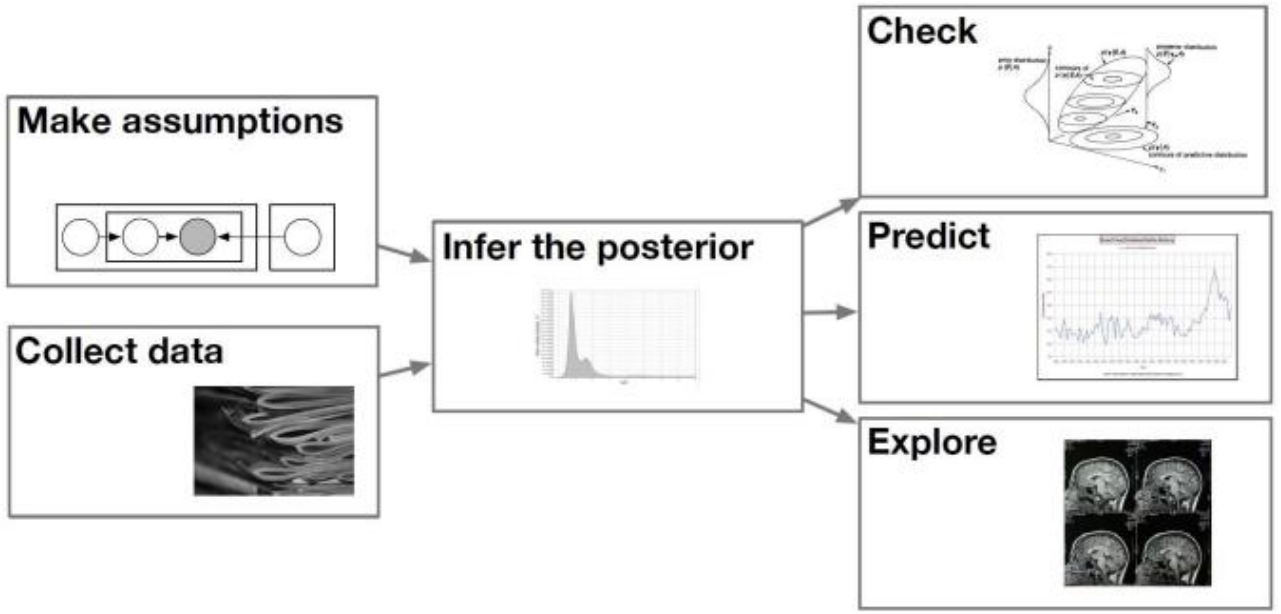
- Topic Model : Generative Approach
  - 데이터를 통해 우리가 가정한 확률적 모델에 잘 맞춰야 한다.
    - ✓ 확률값을 최대로 해야한다.
  - 생성모델은 조건부 확률을 정의한다.P(D|h)
    - ✓ H라는 가설로 D라는 데이터셋이 만들어진다.(Document의 집합)
    - ✓ H라는 가정을 추론 하는 것이 '학습'이 된다.(가정으로부터 데이터셋이 생성 될 확률을 가장 크게 만드는 하이퍼 파라미터를 찾는 것을 의미한다.)





Topic Model

- Topic Model : Generative Approach
  - 데이터를 통해 우리가 가정한 확률적 모델에 잘 맞춰야 한다.
    - ✓ 확률값을 최대로 해야한다.
  - 생성모델은 조건부 확률을 정의한다.P(D|h)
    - ✓ H라는 가설로 D라는 데이터셋이 만들어진다.
    - ✓ H라는 가정을 추론 하는 것이 '학습'이 된다.



02

Probabilistic Latent Semantic Analysis

Probabilistic Latent Semantic Analysis

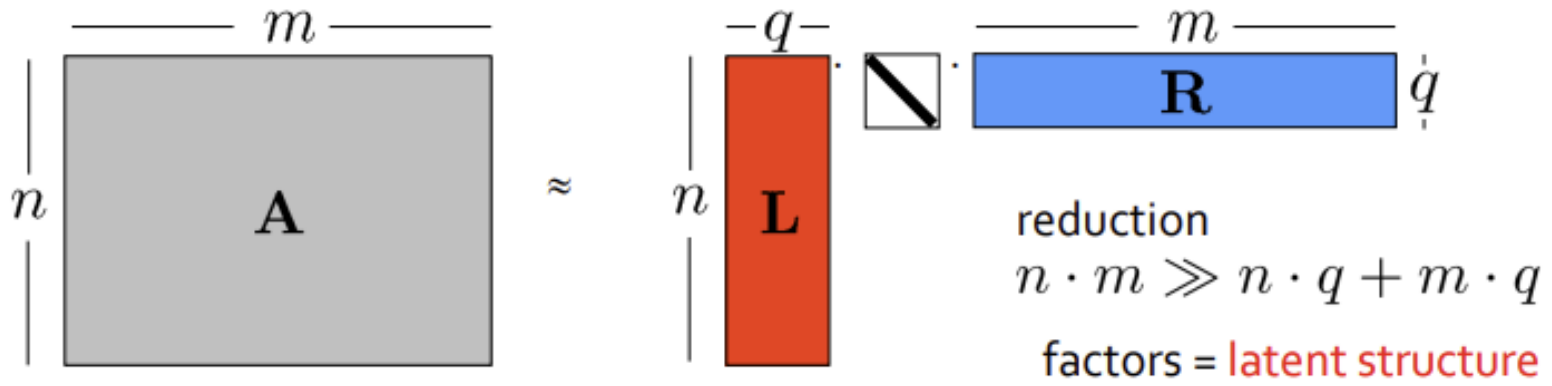
- PLSA
  - 모델의 토픽 분포를 찾기 위함이다.
- Latent Structure
  - 단어, 문장의 등장 빈도를 행렬로 나타내어 벡터로 표
    - ✓ 너무 크고, 복잡하고, 구조가 없다는 문제점이 있다.
    - ✓ TDM,TF-IDF등이 있다

$$\mathbf{A} = \begin{pmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1m} \\ \dots & \dots & \dots & \dots & \dots \\ a_{i1} & \dots & a_{ij} & \dots & a_{im} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & \dots & a_{nj} & \dots & a_{nm} \end{pmatrix}$$

  - 문제점을 해결하기 위해, 쉽고 단순하게 정리 하는 방법을 찾는다.
    - ✓ 이는 잠재의미 분석을 통해 가능하다.

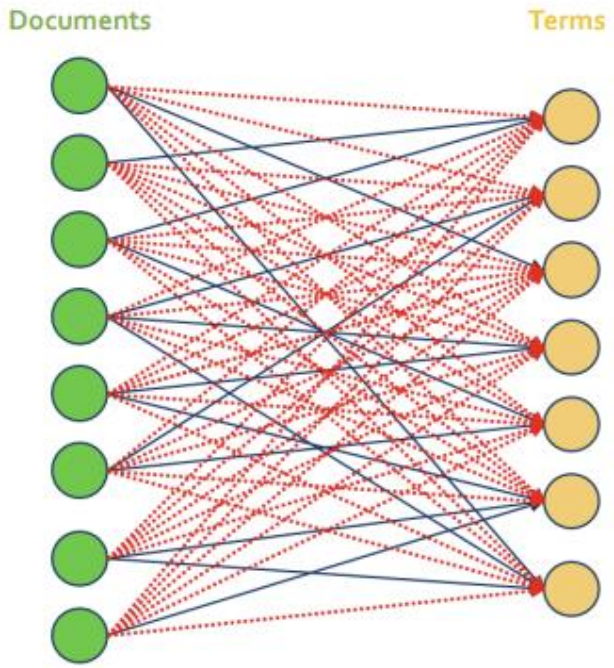
Probabilistic Latent Semantic Analysis

- Latent Structure
  - 단어, 문장의 등장 빈도를 행렬로 나타내어 벡터로 표현할 수 있지만 차원이 너무 크다.
    - ✓ 너무 크고, 복잡하고, 구조가 없다는 문제점이 있다.
    - ✓ TDM,TF-IDF등이 있다
  - 특이값 분해를 이용한다.
    - ✓ A라는 큰 행렬은 L 시그마 R로 분해가 가능하다.
    - ✓ 이때 q는 latent structure(토픽) 이라고 한다.



Probabilistic Latent Semantic Analysis

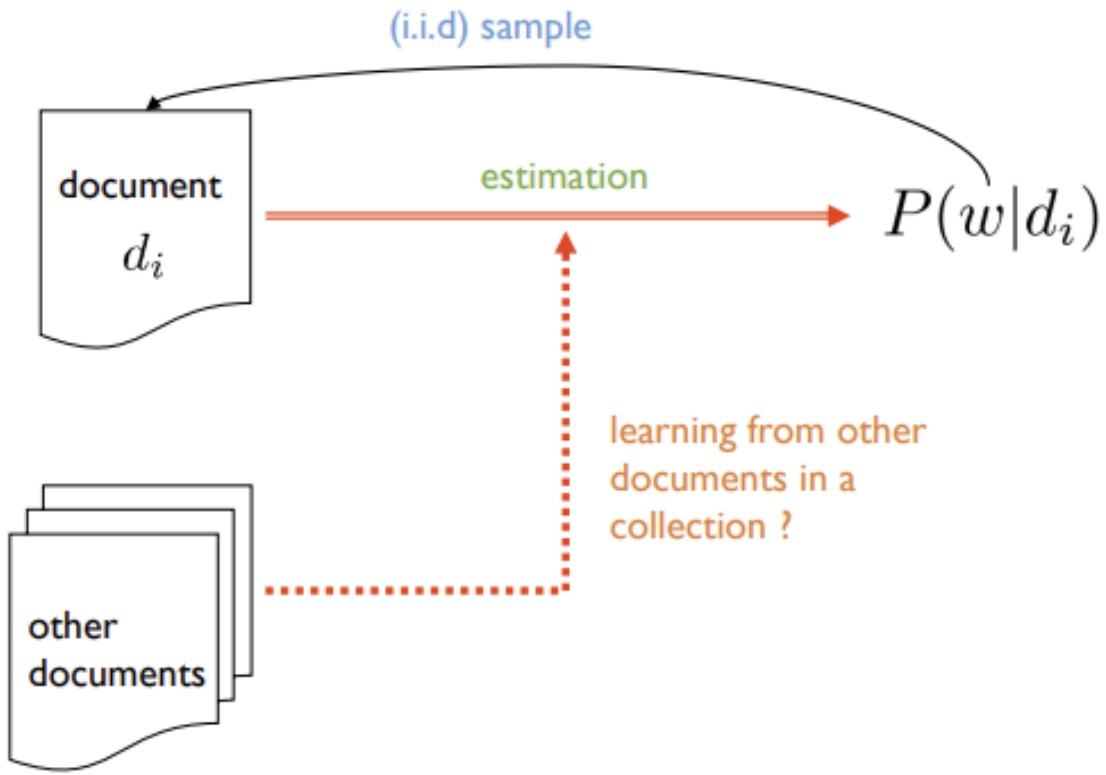
- Language Model : Naïve Approach
  - Maximum likelihood estimation 을 이용한다.
    - ✓ W라는 단어가 d라는 문서에 등장할 확률은 모든 문서에 대한 단어의 수를 분모, 해당하는 문서에 단어가 등장하는 수를 분자로 나타내면 된다.
      - ❖ 문서에 존재하지 않는 단어는 0의 값을 갖는 문제가 있다.



$$\hat{P}_{\mathrm{ML}}(w|d) = \frac{n(d,w)}{\sum_{w'} n(d,w')}$$

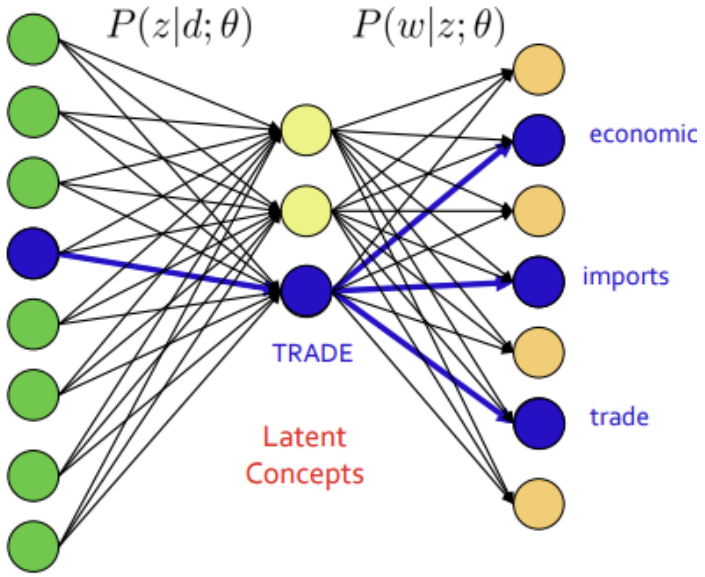
Probabilistic Latent Semantic Analysis

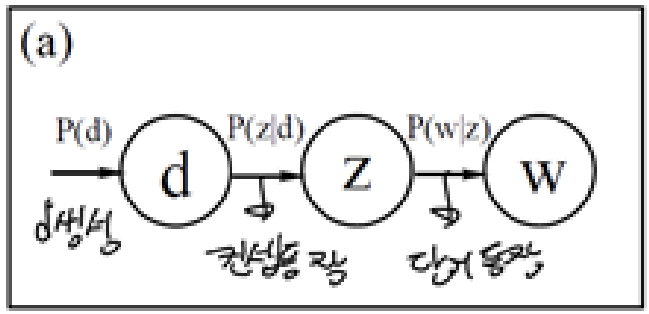
- Language Model : Estimation Problem
  - 다른 문서들의 정보를 이용하면 스무딩이 되지 않을까? 라는 가정을 세운다.

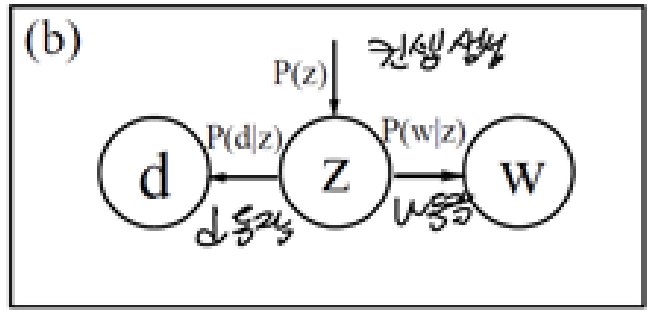


Probabilistic Latent Semantic Analysis

- PLSA
  - Z라는 latent vector 를 만든다.
  - 문서에서 컨셉을 통한 뒤 단어로 간다.
    - ✓ 이때, 컨셉간 중요도는 없고 오직 동시 발생 빈도만 본다
    - ✓ 문서가 있을 때 단어가 발생 확률 = 문서존재 했을 때 컨셉이 나올 확률과 컨셉 안에 단어가 등장 확률을 전부 곱하면 된다..





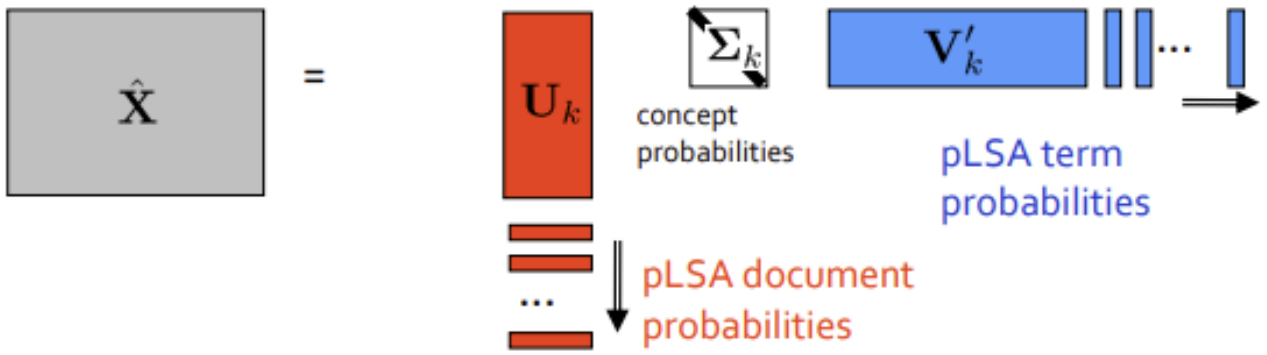


$$\hat{P}_{\mathrm{LSA}}(w|d) = \sum_{z} P(w|z;\theta)P(z|d;\pi)$$

Probabilistic Latent Semantic Analysis

- PLSA
  - Z라는 latent vector 를 만든다.
  - 문서에서 컨셉을 통한 뒤 단어로 간다.
    - ✓ 이때, 컨셉간 중요도는 없고 오직 동시 발생 빈도만 본다
    - ✓ 문서가 있을 때 단어가 발생 확률 = 문서존재 했을 때 컨셉이 나올 확률과 컨셉 안에 단어가 등장 확률을 전부 곱하면 된다..

$$\hat{P}_{\mathrm{LSA}}(d,w) = \sum_z P(d|z)\,P(z)\,P(w|z) \quad = P(d)\sum_z P(w|z)P(z|d)$$



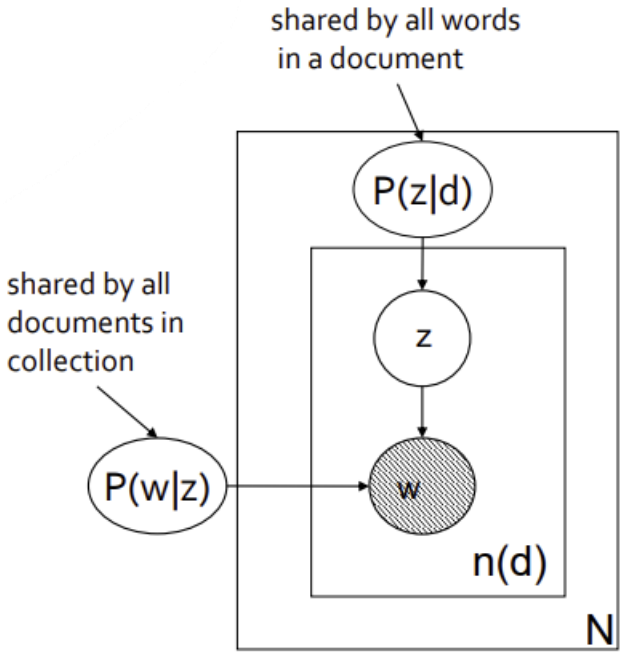
Probabilistic Latent Semantic Analysis

- PLSA
  - Graphical Representation

$$P(w|d) = \sum_z P(w|z)P(z|d)$$

  - P(z|d)
    - ✓ 문서 안에서는 등장 확률이 동일하다.
    - ✓ 단 문서가 달라지면 등장 확률이 달라진다.
      - ❖ Ex) 1번 문서 토픽 40%, 2번 문서 토픽 20%
  - P(w|z)
    - ✓ 문서마다 토픽이 주어졌을 때 단어의 등장 확률을 의미한다.
      - ❖ 이는 문서에 귀속되지 않는다.
      - ❖ IT라는 토픽에 Computer라는 단어가 있을 때

      Computer라는 단어를 뽑을 확률은 1번 문서와 2번 문서가 동일하다.
  - Z
    - ✓ Z가 현재 문서에서 차지하는 비중을 의미한다.
  - W
    - ✓ Z가 현재 문서에서 차지하는 비중+Z에서 단어가 뽑힐 확률에 영향을 받아 만들어진다.



Probabilistic Latent Semantic Analysis

- PLSA Parameter inference
  - 파라미터를 Maximum Likelihood Estimator를 통해 추정한다.
  - 먼저 Likelihood function 을 정의한다.
  - $n(w_i, d_j)$ 은 문서안에 단어가 나타날 빈도 수를 의미한다.
  - $p(w_i, d_j)$ 는 문서안에 단어가 나타날 확률을 의미한다.

  - 따라서, 단어를 문서안에서 n번만큼 볼 확률은 $p(w_i, d_j)$^n이 된다.

Probabilistic Latent Semantic Analysis

- PLSA Parameter inference
  - 모든 단어의 생성은 독립을 가정한다.

$$L = \prod_{i=1}^{m} \prod_{j=1}^{n} p(w_i, d_j)^{n(w_i, d_j)}$$

  - 로그를 씌운다.
    - ✓ 로그 안쪽은 결합 확률이다.

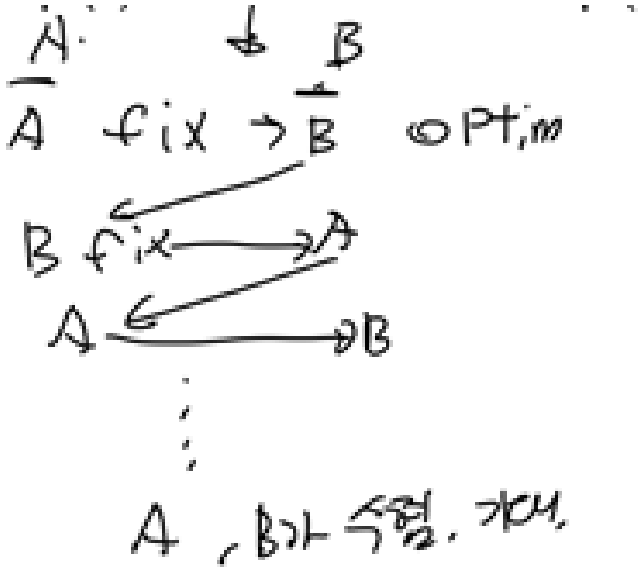
$$\mathcal{L} = \sum_{i=1}^{m} \sum_{j=1}^{n} n(w_i, d_j) log\big(p(w_i, d_j)\big)$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{n} n(w_i, d_j) log\Big(\sum_{l=1}^{k} p(w_i|z_l) p(z_l) p(d_j|z_l)\Big)$$

$$\hat{P}_{\mathrm{LSA}}(d, w) = \sum_{z} P(d|z)\, P(z)\, P(w|z) \quad = P(d) \sum_{z} P(w|z) P(z|d)$$

Probabilistic Latent Semantic Analysis

- PLSA Parameter inference
  - EM알고리즘(Expectation-Maximization)을 사용한다.
    - ✓ A와 B가 있을 때 A를 fix하고 B를 A에 맞게 최적화 한다.
    - ✓ 이후 B를 fix하고 A를 B에 맞게 최적화 한다.
    - ✓ 이걸 반복하면 A와 B가 수렴 할 것을 기대한다.



Probabilistic Latent Semantic Analysis

- PLSA Parameter inference
  - EM알고리즘(Expectation-Maximization)을 사용한다.
    - ✓ A와 B가 있을 때 A를 fix하고 B를 A에 맞게 최적화 한다.
    - ✓ 이후 B를 fix하고 A를 B에 맞게 최적화 한다.
    - ✓ 이걸 반복하면 A와 B가 수렴 할 것을 기대한다.
      - ❖ d라는 문서와 w라는 단어가 주어졌을 때 토픽이 차지하는 비율을 계산 하려 한다.
        - » 분모는 일반화를 위함이다.

$$p(z|d,w) = \frac{P(z)P(d|z)P(w|z)}{\sum_{z' \in Z} P(z')P(d|z')P(w|z')}$$

Probability that the occurence of term w in document d can be "explained" by concept z

$$P(w|z) = \frac{\sum_{d \in D} n(d,w)P(z|d,w)}{\sum_{d \in D, w' \in W} n(d,w')P(z|d,w')}$$

how often is term w associated with concept z ?

$$P(d|z) = \frac{\sum_{w \in W} n(d,w)P(z|d,w)}{\sum_{d' \in D, w \in W} n(d',w)P(z|d',w)}$$

how often is document d associated with concept z ?

$$P(z) = \frac{\sum_{d \in D, w \in W} n(d,w)P(z|d,w)}{\sum_{d \in D, w \in W} n(d,w)}$$

how prevalent is the concept z ?

Probabilistic Latent Semantic Analysis

- PLSA example

Raw Data

	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6
Ba						
Bas						
B						
M						
In						
De						
Rep						
C						
Pr						

Initialization

Topic 1	Topic 2	Topic 3
0.525	0.407	0.068

	Topic 1	Topic 2	Topic 3
Doc 1	0.020	0.008	0.048
Doc 2	0.294	0.255	0.329
Doc 3	0.204	0.138	0.178
Doc 4	0.200	0.146	0.007
Doc 5	0.186	0.196	0.233
Doc 6	0.096	0.257	0.205

$P(z)$

Topic 1	Topic 2	Topic 3
0.525	0.407	0.068

After 1 EM step

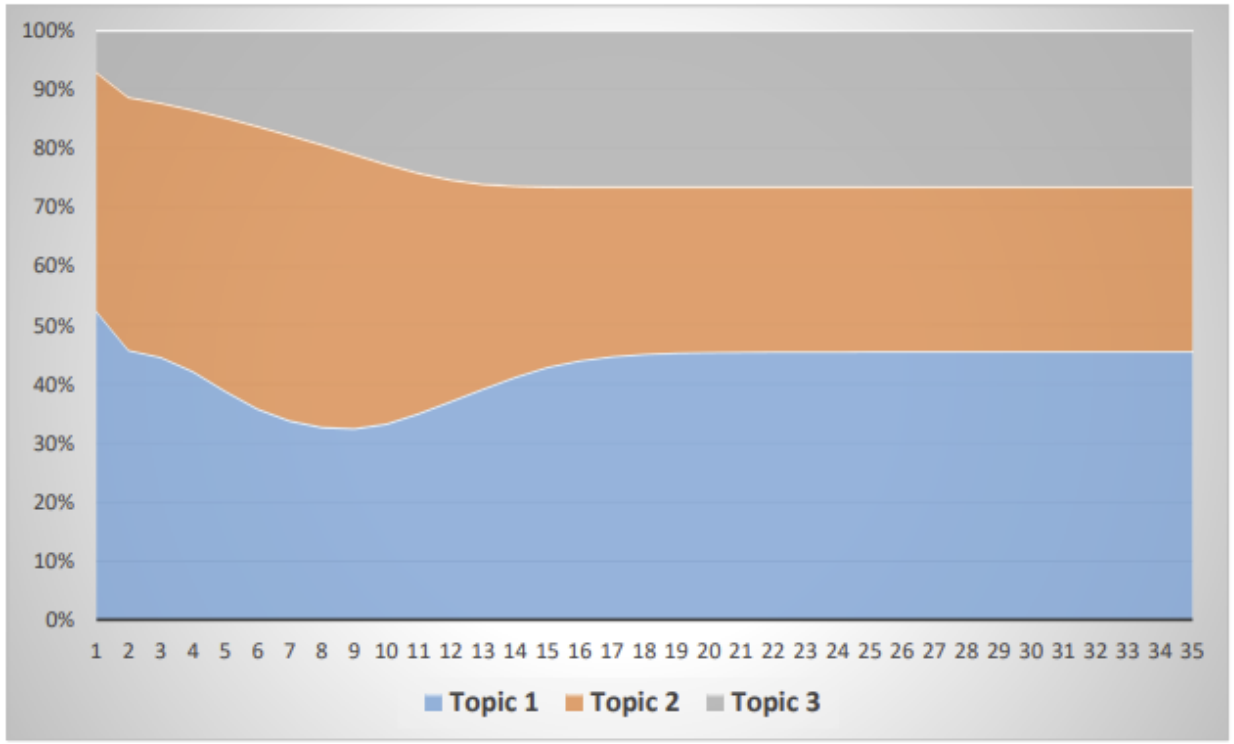
Topic 1	Topic 2	Topic 3
0.459	0.430	0.111

	Topic 1	Topic 2	Topic 3
Doc 1	0.180	0.077	0.382
Doc 2	0.124	0.089	0.091
Doc 3	0.147	0.213	0.149
Doc 4	0.125	0.110	0.004
Doc 5	0.266	0.204	0.167
Doc 6	0.158	0.308	0.207

1	Topic 2	Topic 3
	0.016	0.010
	0.133	0.166
	0.058	0.133
	0.088	0.145
	0.030	0.044
	0.167	0.056
	0.129	0.201
	0.156	0.039
	0.114	0.008
	0.110	0.199

Probabilistic Latent Semantic Analysis

- PLSA example



Probabilistic Latent Semantic Analysis

- PLSA example

	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6
Baseball	1	2	0	0	0	0
Basketball	3	1	0	0	0	0
Boxing	2	0	0	0	0	0
Money	3	3	2	3	2	4
Interest	0	0	3	2	0	0
Rate	0	0	4	1	0	0
Democrat	0	0	0	0	4	3
Republican	0	0	0	0	2	1
Cocus	0	0	0	0	3	2
President	0	0	1	0	2	3

Topic 1	Topic 2	Topic 3
0.456	0.281	0.263

	Topic 1	Topic 2	Topic 3
Doc 1	0.000	0.000	0.600
Doc 2	0.000	0.000	0.400
Doc 3	0.000	0.625	0.000
Doc 4	0.000	0.375	0.000
Doc 5	0.500	0.000	0.000
Doc 6	0.500	0.000	0.000

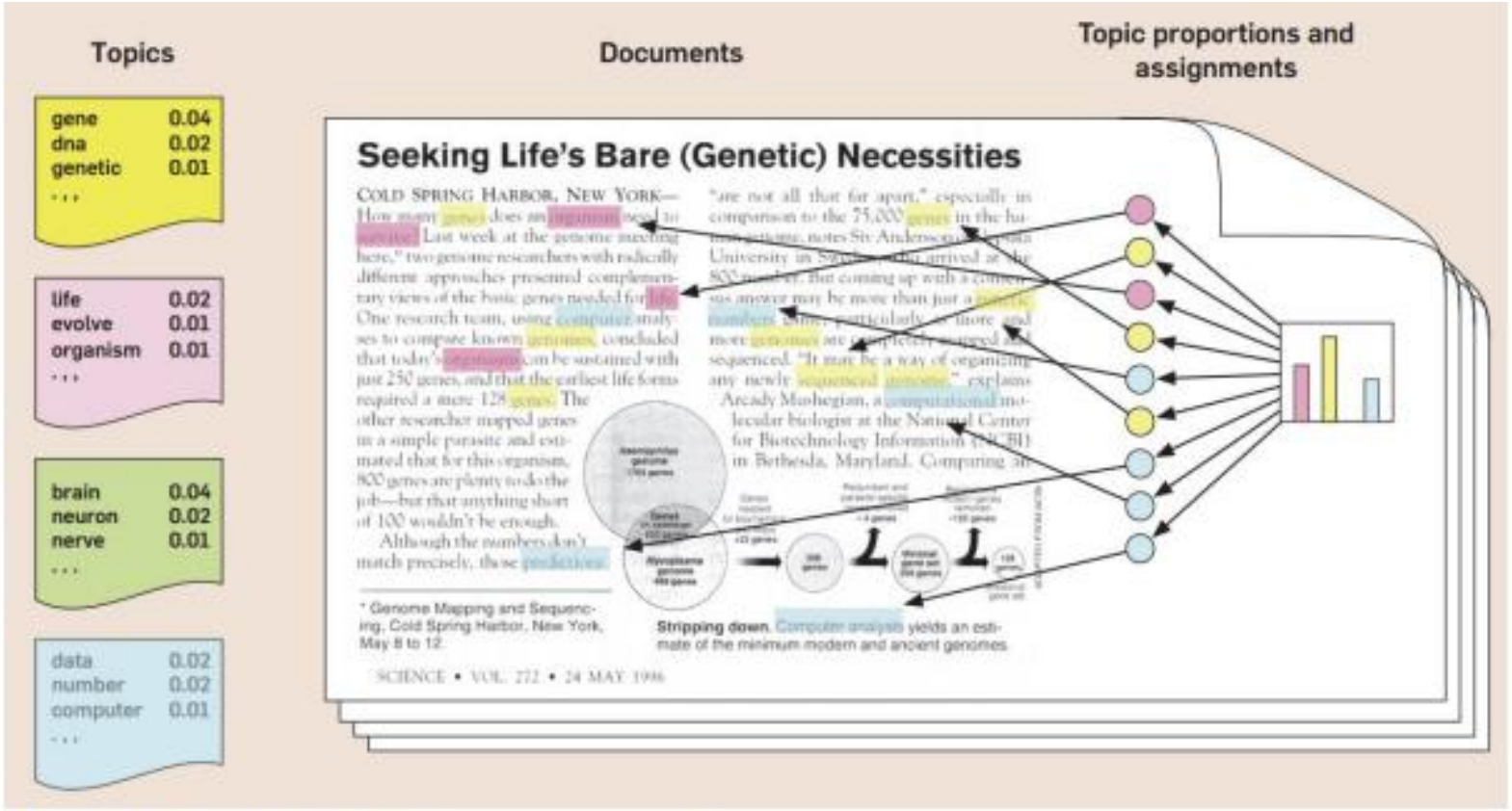
	Topic 1	Topic 2	Topic 3
Baseball	0.000	0.000	0.200
Basketball	0.000	0.000	0.267
Boxing	0.000	0.000	0.133
Money	0.231	0.313	0.400
Interest	0.000	0.312	0.000
Rate	0.000	0.312	0.000
Democrat	0.269	0.000	0.000
Republican	0.115	0.000	0.000
Cocus	0.192	0.000	0.000
President	0.192	0.063	0.000

03

LDA

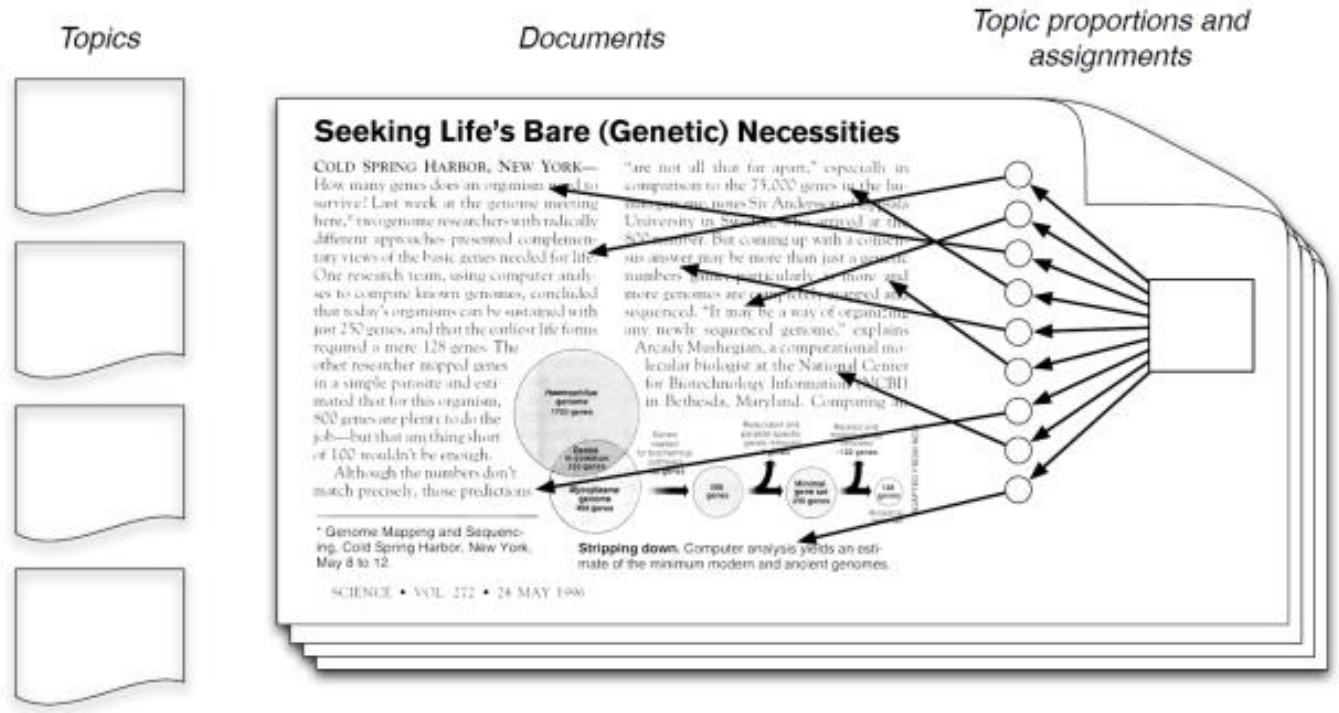
Document Generation Process

- LDA : Intuition
  - 문서 생성을 위한 방법이다.
    - ✓ 토픽, 단어, 문장의 토픽 빈도는 다 파라미터 이다.



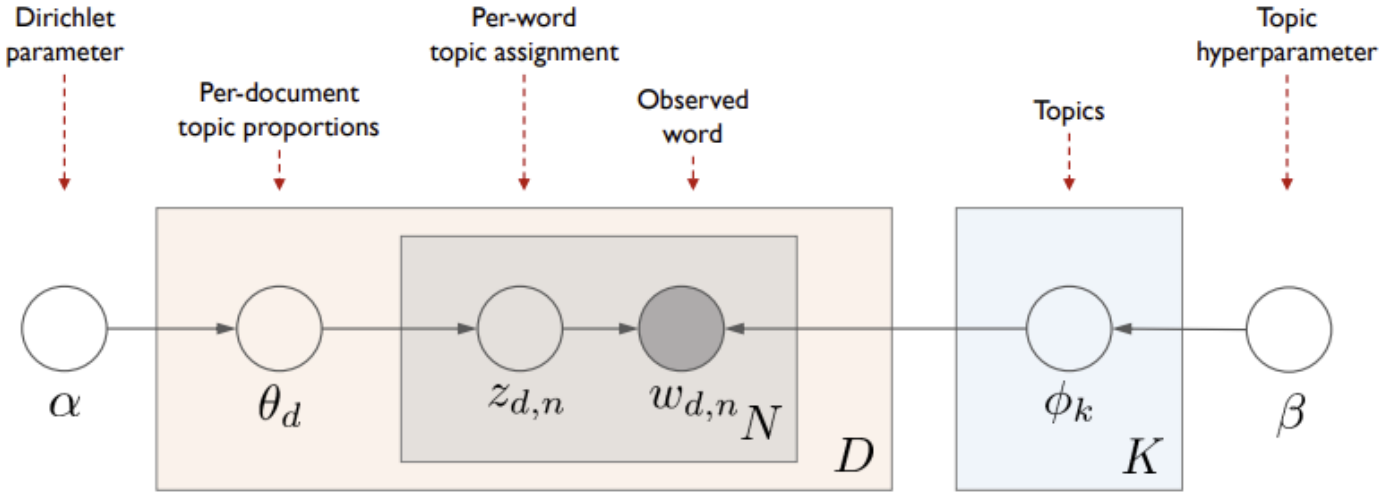
Document Generation Process

- LDA : Intuition
  - 문서 생성을 위한 방법이다.
    - ✓ 토픽, 단어, 문장의 토픽 빈도는 다 변수 이다.



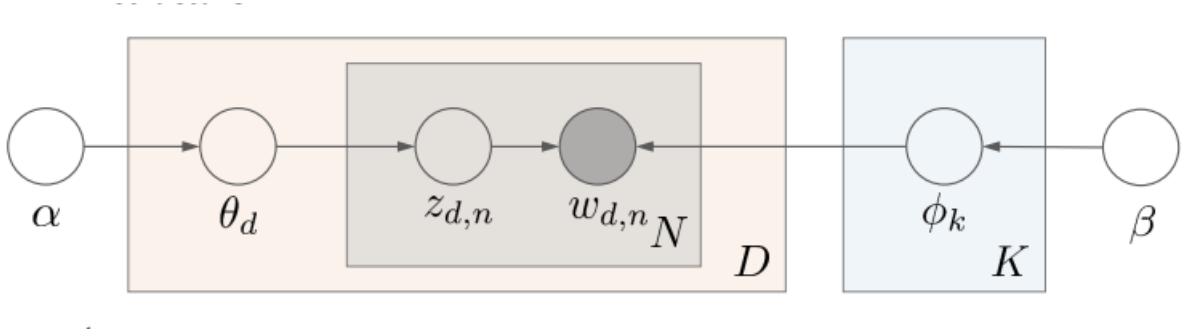
Document Generation Process

- LDA Overview
  - 모든 단어에 대해 독립을 가정한다.
  - 하나의 문서는 여러 개의 토픽을 갖는다.
  - 알파와 베타는 하이퍼 파라미터 이다.
  - 세타는 문서별로 몇 개의 토픽에 얼마나 분포되어 있는지를 의미한다.
  - 파이는 토픽에 대한 단어의 분포를 의미한다.
  - Z(d,n)은 d번째 문서에서 n번째 단어가 어디 토픽에서 왔는지를 의미한다.



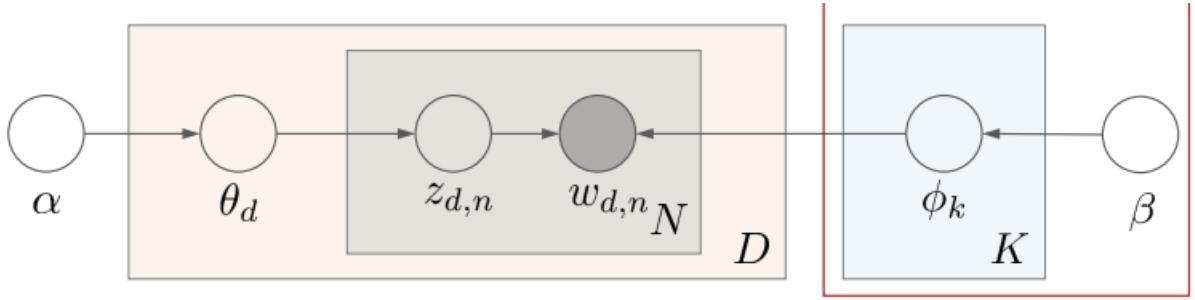
Document Generation Process

- LDA structure
  - 각각의 토픽을 베타를 하이퍼 파라미터로 갖는 디리슈레 분포로 추정한다.
  - N은 문서마다 다른 단어의 수를 의미한다.
  - D는 문서의 수를 의미한다.
  - K는 토픽의 수를 의미한다.



Document Generation Process

- LDA structure
  - 각각의 토픽을 베타를 하이퍼 파라미터로 갖는 디리슈레 분포로 추정한다.
  - N은 문서마다 다른 단어의 수를 의미한다.
  - D는 문서의 수를 의미한다.
  - K는 토픽의 수를 의미한다.



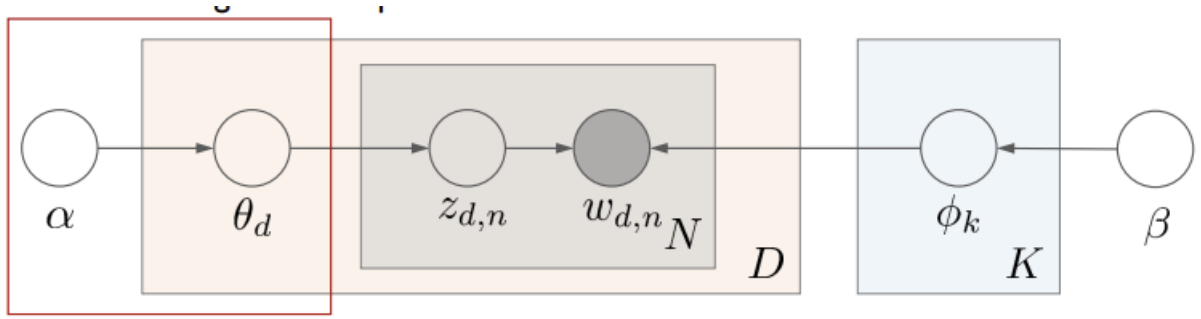
  - 베타에대한 파이값을 조건부 확률로 나타낼 수 있다.

$$p(\phi|\beta) = \prod_{k=1}^{K} \frac{\Gamma(\beta_{k,\cdot})}{\prod_{v=1}^{V} \Gamma(\beta_{k,v})} \prod_{v=1}^{V} \phi_{k,v}^{\beta_{k,v}-1}$$

	Term 1	Term 2	Term 3	Term 4	Term 5=V
Topic 1 $\phi_{k=1}$	0.1	0.1	0	0.7	0.1
Topic 2 $\phi_{k=2}$	0.2	0.1	0.2	0.2	0.3
Topic 3 $\phi_{k=3}$	0.01	0.2	0.39	0.3	0.1
Topic 4 $\phi_{k=4=K}$	0.0	0.0	0.5	0.3	0.2

Document Generation Process

- LDA structure
  - 각각의 토픽을 베타를 하이퍼 파라미터로 갖는 디리슈레 분포로 추정한다.
  - N은 문서마다 다른 단어의 수를 의미한다.
  - D는 문서의 수를 의미한다.
  - K는 토픽의 수를 의미한다.

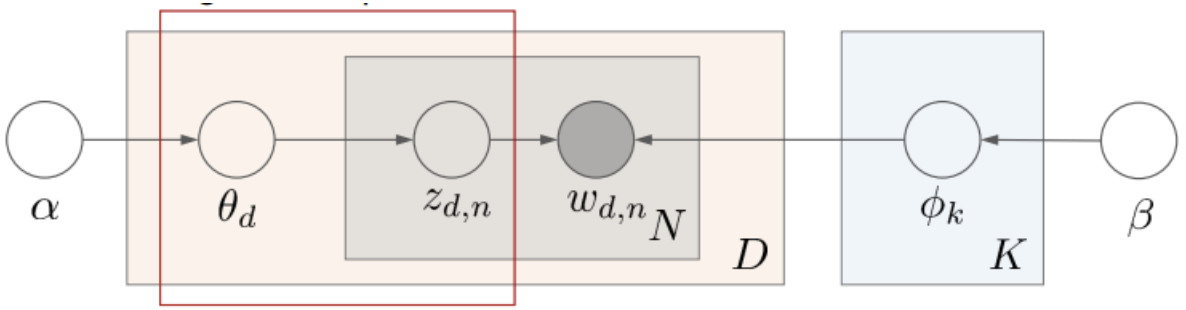


$$p(\theta|\alpha) = \frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\theta_1^{\alpha_k-1}\cdots\theta_K^{\alpha_K-1} = \frac{\Gamma(\alpha_.)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\prod_{k=1}^{K}\theta_k^{\alpha_k-1}$$

		Topic 1	Topic 2	Topic 3	Topic 4
Document 1	$\theta_{d=1}$	0.5	0.1	0.3	0.1
Document 2	$\theta_{d=2}$	0.0	0.9	0.1	0.0
Document 3	$\theta_{d=3=D}$	0.02	0.48	0.25	0.25

Document Generation Process

- LDA structure
  - 각각의 토픽을 베타를 하이퍼 파라미터로 갖는 디리슈레 분포로 추정한다.
  - 세타는 문서별로 몇 개의 토픽에 얼마나 분포되어 있는지를 의미한다.
  - 파이는 토픽에 대한 단어의 분포를 의미한다.
  - Z(d,n)은 d번째 문서에서 n번째 단어가 어디 토픽에서 왔는지를 의미한다.

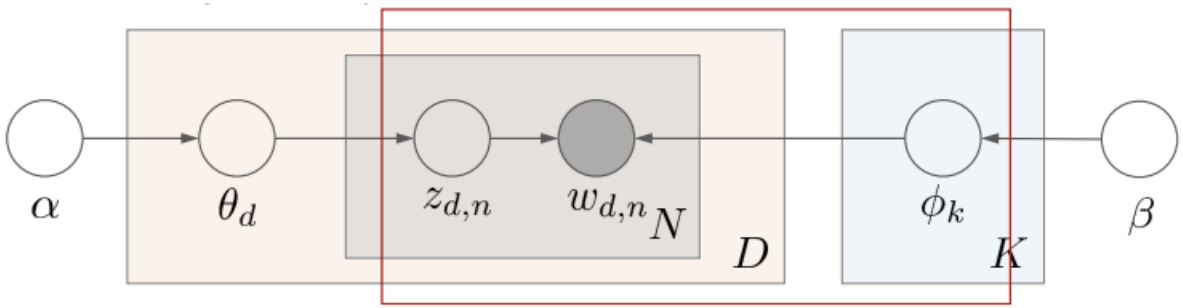


		Word w_1	Word w_2	Word w_3	Word w_4	Word w_5	Word w_6
Document 1	$z_{d=1}$	Topic k=2	Topic k=1	Topic k=1	Topic k=4	Topic k=3	Topic k=3
Document 2	$z_{d=2}$	Topic k=2	Topic k=3	Topic k=2	Topic k=2		
Document 3	$z_{d=3=D}$	Topic k=4	Topic k=2	Topic k=2	Topic k=4	Topic k=3	

$$p(z|\theta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k,.}}$$

Document Generation Process

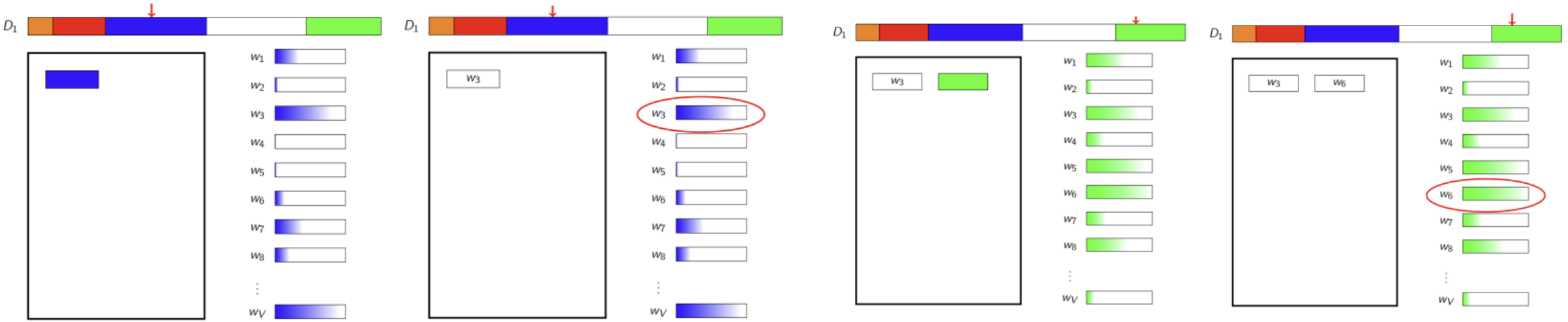
- LDA structure
  - 각각의 토픽을 베타를 하이퍼 파라미터로 갖는 디리슈레 분포로 추정한다.
  - 세타는 문서별로 몇 개의 토픽에 얼마나 분포되어 있는지를 의미한다.
  - 파이는 토픽에 대한 단어의 분포를 의미한다.
    - ✓ 코퍼스 안에서를 의미한다.
  - Z(d,n)은 d번째 문서에서 n번째 단어가 어디 토픽에서 왔는지를 의미한다.



$$p(w|z,\phi) = \prod_{k=1}^{K} \prod_{v=1}^{V} \phi_{k,v}^{n_{.,k,v}}$$

Document Generation Process

- Document generation process.



04

실습

실습

- Gensim
  - 벡터 변환, 토픽 모델링, 텍스트 요약 등 자연어와 텍스트 마이닝 관련 여러 기능을 제공하는 라이브러리이다.
- 뉴스 그룹 데이터
  - Sklearn에서 제공하는 데이터 셋이다.

```
from sklearn.datasets import fetch_20newsgroups

dataset = fetch_20newsgroups(shuffle=True, random_state=123,
                             remove=('headers','footers','quotes'))
print(len(dataset.data))
print(dataset.target_names)
```

```
11314
['alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.windows.x', 'misc.forsale', 'rec.autos', 'rec.motorcycles',
```

```
dataset.data[1]
```

```
'From article <1993Apr18.000152.2339@gnv.ifas.ufl.edu>, by jrm@gnv.ifas.ufl.edu:₩n₩nOnly irrational fools such as yourself are set against RKBA.  There are₩n*plenty* of people who support it.₩n₩n
₩nThe government will be overthrown *long* before that happens.  A *huge* ₩nmillitia composed of all available men and women who care about their₩ncountry will defeat the forces of the evil Klinto
nistas.  The people₩n*will* prevail!₩n₩nOh, so you think armed citizens alone can₩'t overthrow the government?₩nConsider this:  do you think *all* law enforcement officials and members₩nof the Arm
ed Forces will turn against the people that they are entrusted₩nto serve?  Not hardly.  You can count on a lot of people in the Army,₩nMarines, Air Force, Navy, National Guard, police officers, an
d so on₩njoining in the cause to defend the liberties and freedoms of American₩ncitizens.  COUNT ON IT!  THE GOVERNMENT WILL NOT BE ABLE TO DISARM₩nEVERYONE WITHOUT STARTING A CIVIL WAR!₩n₩n₩nWron
g again.  People will ...'
```

실습

- Preprocessing

```
import pandas as pd
news_df = pd.DataFrame({'article' : dataset.data})
```

```
news_df['article']
```

```
0        A quick query for Powerbook gurus:₩nTom Spearm...
1        From article <1993Apr18.000152.2339@gnv.ifas.u...
2        ₩nSteve, take a look at what you are saying.  ...
3        I have a routine that changes the color (RGB) ...
4        ₩nI sometimes wonder if Kekule's dream wasn't ...
                               ...
11309    ₩n₩ni would like to remind my jewish colleague...
11310    ₩nProbably because everyone (that is, everyone...
11311    THere is a defect in the 13" hi-res monitors, ...
11312    ₩na lot of batters lean in when pitches come. ...
11313    Howdy,₩n₩nI'm a little new to this newsgroup, ...
Name: article, Length: 11096, dtype: object
```

```
0        quick query powerbook gurustom spearman post a...
1        article jrmgnvifasufleduonly irrational fools ...
2        steve take look saying dont construvtiveword d...
3        routine changes color attributes myvga adapter...
4        sometimes wonder kekules dream wasnt influence...
                               ...
11309    would like remind jewish colleague much thesto...
11310    probably everyone everyone cable watchevery br...
11311    defect hires monitors bring dealer replace fly...
11312    batters lean pitches come rickeys crouch tends...
11313    howdyim little newsgroup would like theknowled...
Name: article, Length: 11096, dtype: object
```

```
def clean_stopword(d):
    stop_words = stopwords.words('english') + ['u','im','c']
    return ' '.join([w.lower()for w in d.split() if w.lower() not in stop_words and len(w) > 3] ) #이거 한줄로 stop word제거 그리고
#세자 이상만 사용 할꺼야
def preprocessing(d):
    return preprocess_string(d)
```

실습

- Tokenizing

```
tokenized_news = news_df['article'] = news_df['article'].apply(preprocessing)
tokenized_news = tokenized_news.to_list()
tokenized_news[1]
#리스트 하나당 토크나이즈 된 것들로 나옴
```

```
['articl',
 'jrmgnvifasufleduonli',
 'irrat',
 'fool',
 'rkba',
 'areplenti',
 'peopl',
 'support',
 'itth',
 'govern',
 'overthrown',
 'long',
 'happen',
 'huge',
 'millitia',
 'compos',
 'avail',
 'women',
 'care',
 'theircountri',
 'defeat',
 'forc',
 'evil',
 'klintonista',
 'peoplewil',
```

```
0        quick query powerbook gurustom spearman post a...
1        article jrmgnvifasufleduonly irrational fools ...
2        steve take look saying dont construvtiveword d...
3        routine changes color attributes myvga adapter...
4        sometimes wonder kekules dream wasnt influence...
                               ...
11309    would like remind jewish colleague much thesto...
11310    probably everyone everyone cable watchevery br...
11311    defect hires monitors bring dealer replace fly...
11312    batters lean pitches come rickeys crouch tends...
11313    howdyim little newsgroup would like theknowled...
Name: article, Length: 11096, dtype: object
```

실습

- Dictionary & Corpus

```
from gensim.corpora.dictionary import Dictionary

dictionary = Dictionary(tokenized_news)
print(len(dictionary))

dictionary.filter_extremes(keep_n= 2000, no_below = 5, no_above = 0.5) #빈도가 너무 적거나 높은 단어들은 제거한다.
print(len(dictionary))
corpus = [dictionary.doc2bow(text) for text in tokenized_news]#bow형태로
print(len(corpus))
```

```
122159
2000
11096
```

실습

- 잠재 의미 분석

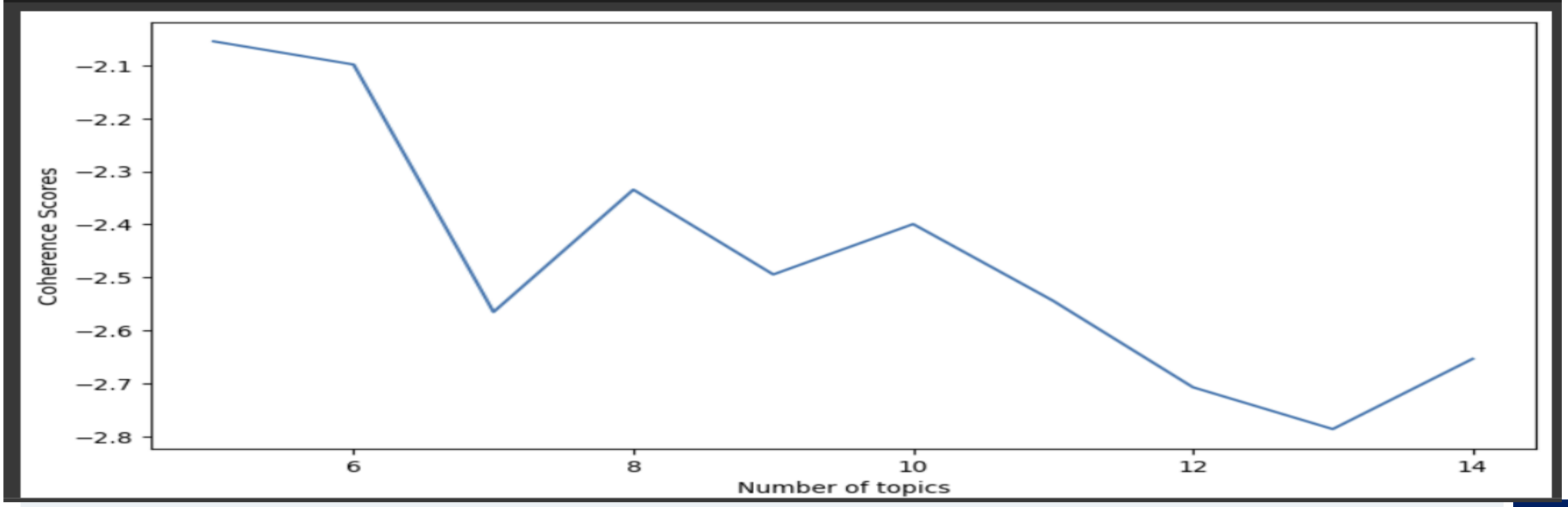
```
from gensim.models import LsiModel
lsi_model = LsiModel(corpus, num_topics = 10, id2word = dictionary)
lsi_model.print_topics(num_words = 10)
```

```
[(0,
  '0.245*"file" + 0.192*"program" + 0.154*"imag" + 0.152*"peopl" + 0.135*"avail" + 0.134*"dont" + 0.132*"work" + 0.130*"know" + 0.128*"like" + 0.126*"includ"'),
 (1,
  '0.284*"file" + -0.229*"know" + -0.228*"peopl" + -0.220*"said" + 0.210*"imag" + -0.187*"dont" + -0.172*"think" + 0.155*"program" + -0.142*"go" + -0.140*"didnt"'),
 (2,
  '-0.702*"wire" + -0.273*"ground" + -0.216*"circuit" + -0.175*"connect" + -0.158*"neutral" + -0.131*"cabl" + -0.123*"electr" + -0.117*"instal" + -0.099*"requir" + -0.098*"usual"'),
 (3,
  '-0.467*"entri" + 0.322*"anonym" + -0.315*"file" + -0.202*"program" + 0.166*"post" + 0.136*"privaci" + 0.136*"internet" + -0.129*"rule" + -0.127*"output" + 0.127*"user"'),
 (4,
  '-0.312*"anonym" + 0.300*"drive" + 0.293*"imag" + -0.264*"entri" + 0.187*"disk" + -0.180*"post" + 0.165*"support" + -0.140*"privaci" + -0.126*"file" + -0.122*"internet"'),
 (5,
  '-0.480*"drive" + 0.333*"imag" + -0.275*"disk" + -0.235*"entri" + -0.223*"control" + -0.194*"hard" + -0.173*"bio" + -0.136*"card" + -0.123*"support" + -0.122*"featur"'),
 (6,
  '0.256*"anonym" + -0.222*"team" + 0.208*"drive" + -0.207*"hockei" + -0.200*"launch" + -0.186*"space" + -0.179*"leagu" + -0.165*"game" + -0.141*"year" + 0.139*"peopl"'),
 (7,
  '0.293*"presid" + -0.253*"team" + -0.249*"hockei" + -0.214*"leagu" + -0.212*"game" + 0.156*"think" + -0.153*"season" + 0.149*"administr" + -0.142*"armenian" + 0.141*"program"'),
 (8,
  '0.421*"imag" + -0.312*"widget" + 0.258*"launch" + -0.231*"window" + 0.227*"space" + 0.168*"satellit" + -0.144*"resourc" + -0.143*"applic" + -0.130*"presid" + -0.130*"server"'),
 (9,
  '0.358*"imag" + -0.325*"launch" + -0.278*"space" + 0.219*"presid" + -0.218*"widget" + -0.200*"satellit" + -0.151*"window" + 0.130*"hockei" + 0.116*"team" + -0.113*"applic"')]
```

실습

- Coherence

```
import matplotlib.pyplot as plt
x = [int(i) for i in range(min_topics,max_topics)]
plt.figure(figsize = (10,5))
plt.plot(x, coherence_scores)
plt.xlabel('Number of topics')
plt.ylabel('Coherence Scores')
plt.show()
```



실습

- 잠재 의미 분석

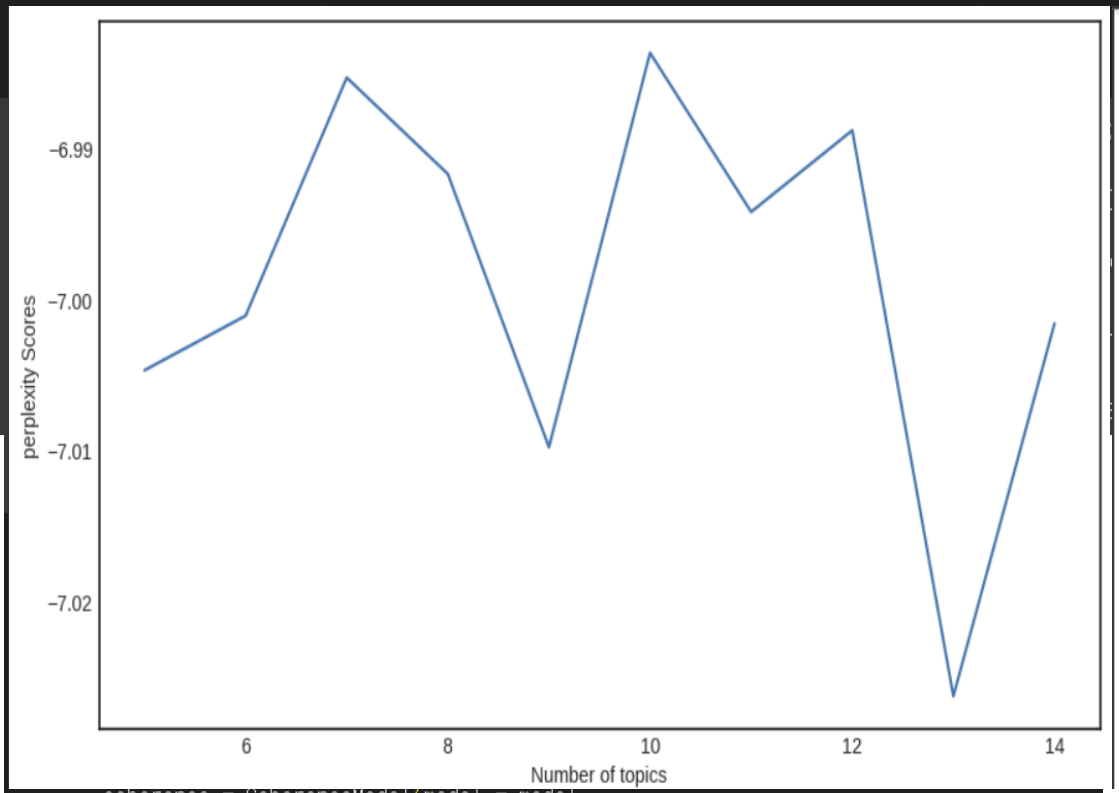
```
lsi_model = LsiModel(corpus, num_topics = 6, id2word = dictionary)
lsi_model.print_topics(num_words = 10)
```

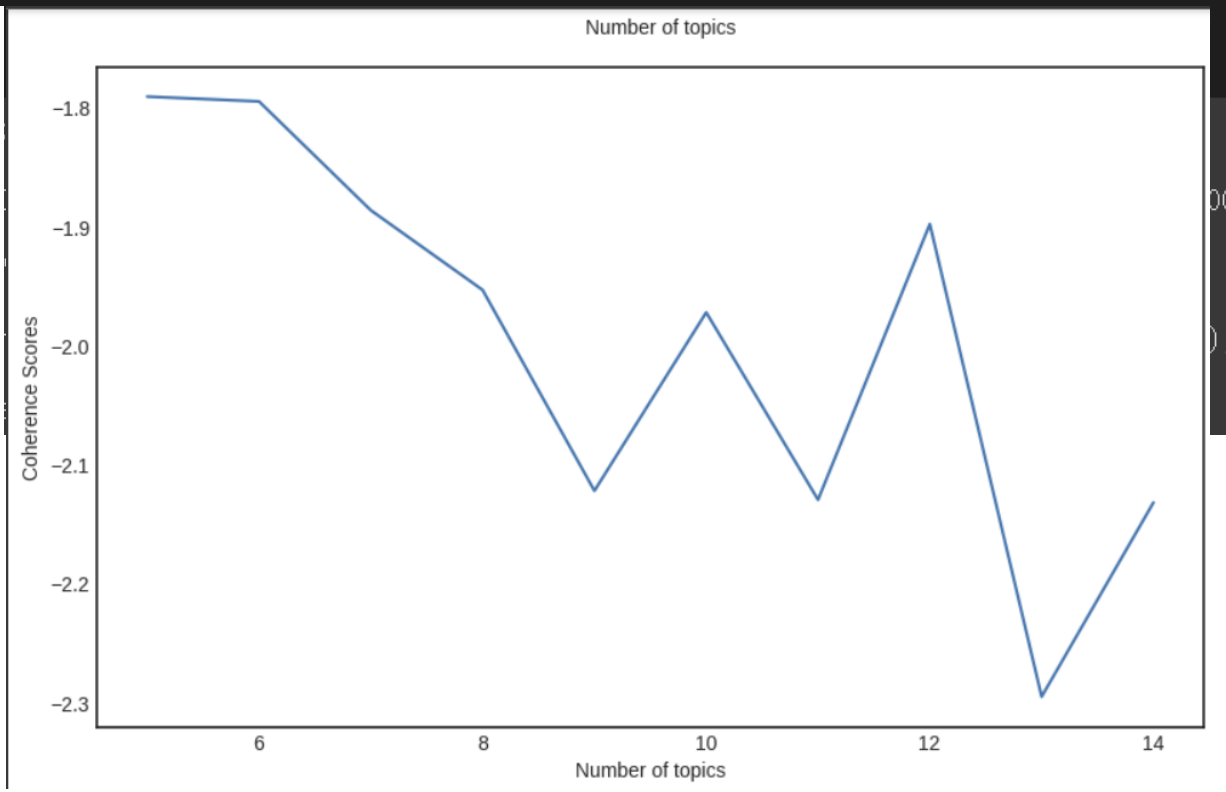
```
[(0,
  '0.245*"file" + 0.192*"program" + 0.154*"imag" + 0.152*"peopl" + 0.135*"avail" + 0.134*"dont" + 0.132*"work" + 0.130*"know" + 0.128*"like" + 0.126*"includ"'),
 (1,
  '0.284*"file" + -0.229*"know" + -0.228*"peopl" + -0.220*"said" + 0.210*"imag" + -0.187*"dont" + -0.172*"think" + 0.155*"program" + -0.142*"go" + -0.140*"didnt"'),
 (2,
  '-0.702*"wire" + -0.273*"ground" + -0.216*"circuit" + -0.175*"connect" + -0.158*"neutral" + -0.131*"cabl" + -0.123*"electr" + -0.117*"instal" + -0.099*"requir" + -0.098*"usual"'),
 (3,
  '0.467*"entri" + -0.322*"anonym" + 0.315*"file" + 0.202*"program" + -0.166*"post" + -0.136*"privaci" + -0.136*"internet" + 0.129*"rule" + 0.127*"output" + -0.127*"user"'),
 (4,
  '0.312*"anonym" + -0.300*"drive" + -0.293*"imag" + 0.264*"entri" + -0.187*"disk" + 0.180*"post" + -0.165*"support" + 0.140*"privaci" + 0.126*"file" + 0.122*"internet"'),
 (5,
  '-0.480*"drive" + 0.333*"imag" + -0.275*"disk" + -0.235*"entri" + -0.223*"control" + -0.194*"hard" + -0.173*"bio" + -0.136*"card" + -0.123*"support" + -0.122*"featur"')]
```

실습

- LDA

```
from gensim.models import LdaModel
```





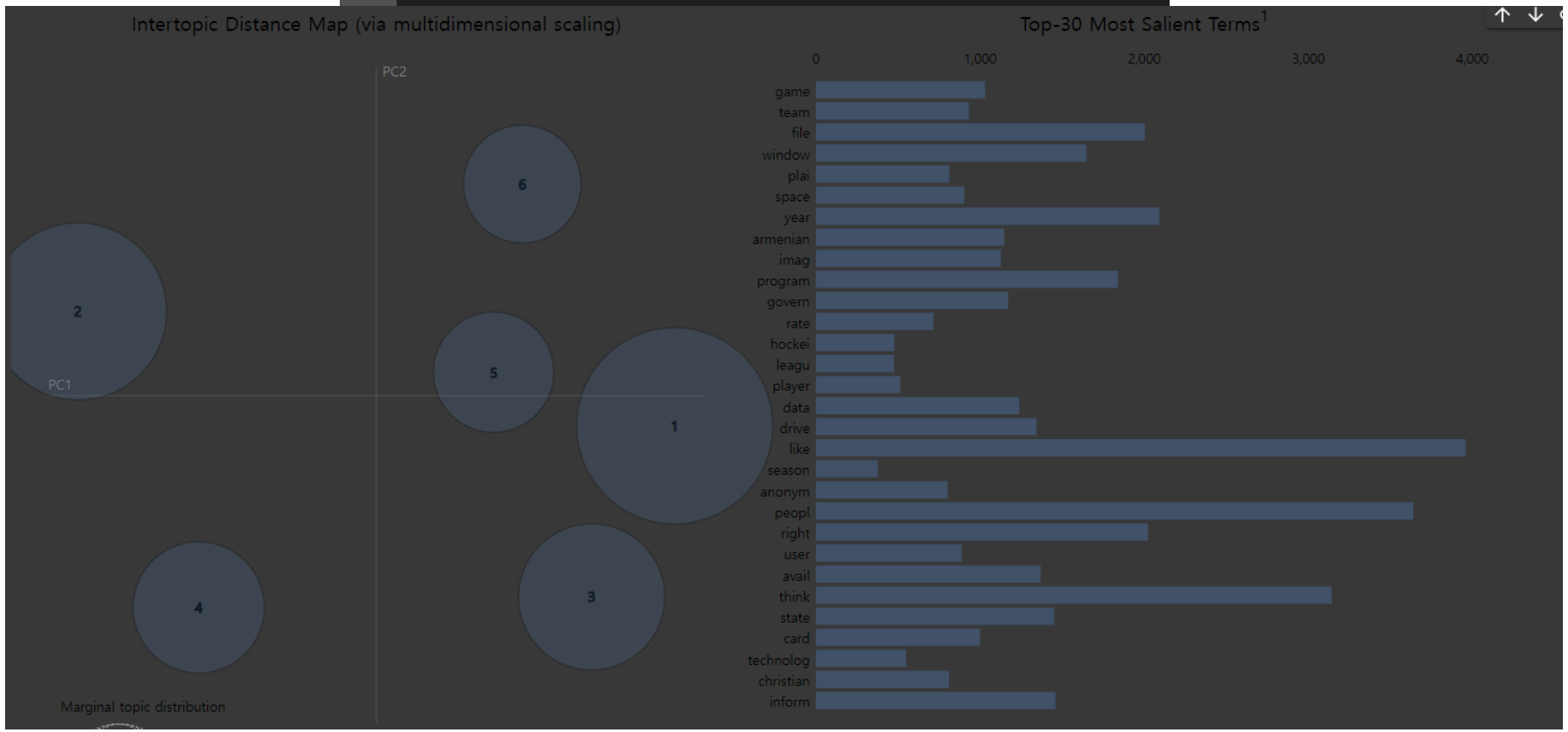

```
coherence = CoherenceModel(model = model,
                           texts =tokenized_news,
                           dictionary = dictionary,
                           coherence = 'u_mass')
coherence_scores.append(coherence.get_coherence())
print(perplexity_values)
print(coherence_scores)
```

실습

- LDA

```
import pyLDAvis
import pyLDAvis.gensim

pyLDAvis.enable_notebook()
vis = pyLDAvis.gensim.prepare(lda_model,corpus,dictionary)
pyLDAvis.display(vis)
```



실습

- LDA

```
def make_topictable_per_doc(lda_model, corpus):
    topic_table = pd.DataFrame()

    # 몇 번째 문서인지를 의미하는 문서 번호와 해당 문서의 토픽 비중을 한 줄씩 꺼내온다.
    for i, topic_list in enumerate(lda_model[corpus]):
        doc = topic_list[0] if lda_model.per_word_topics else topic_list
        doc = sorted(doc, key=lambda x: (x[1]), reverse=True)


        # 모든 문서에 대해서 각각 아래를 수행
        for j, (topic_num, prop_topic) in enumerate(doc): #  몇 번 토픽인지와 비중을 나눠서 저장한다.
```

	문서 번호	가장 비중이 높은 토픽	가장 높은 토픽의 비중	각 토픽의 비중
0	0	4	0.7082	[(4, 0.7082247), (5, 0.258139)]
1	1	2	0.8363	[(2, 0.8362991), (4, 0.15230463)]
2	2	4	0.6840	[(3, 0.28537908), (4, 0.68401414)]
3	3	5	0.9651	[(5, 0.96507764)]
4	4	4	0.8320	[(0, 0.03357397), (1, 0.033660606), (2, 0.0335...
5	5	4	0.5183	[(0, 0.012054871), (1, 0.43352595), (2, 0.0120...