

비정형텍스트분석

6. Word2vec, Doc2vec

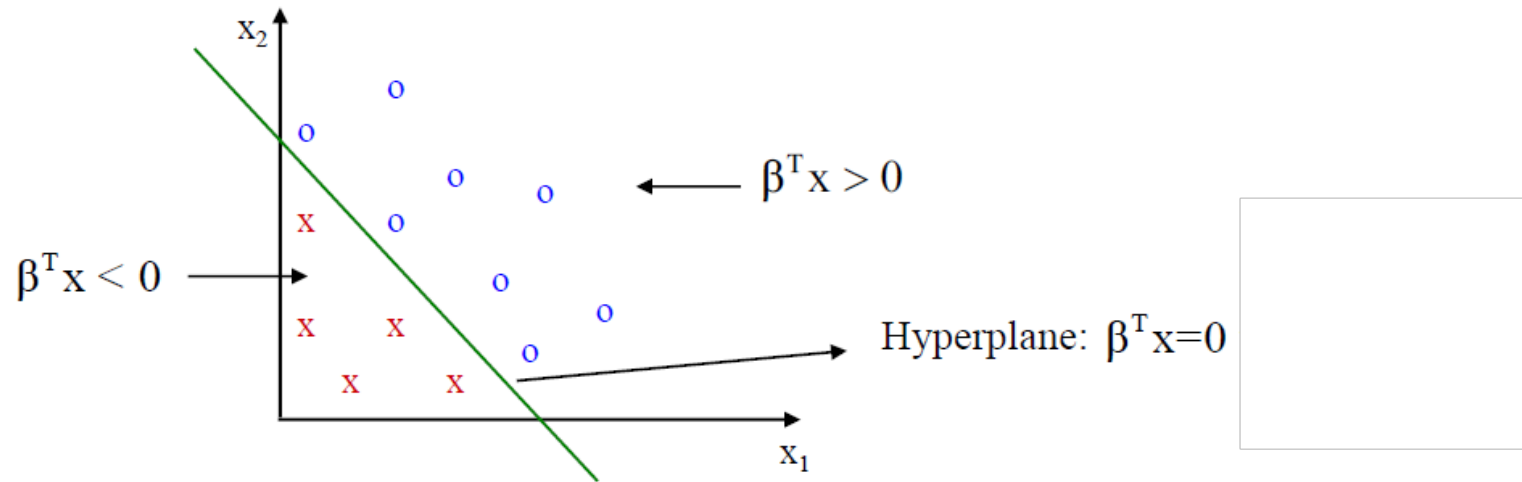
2020년 10월 05일

데이터사이언스학과 / 이영훈

Classifier Basic

Logistic Regression

- Logistic Regression (LR)은 대표적인 binary classification 알고리즘



Classifier

$$y = \frac{1}{(1 + \exp(-\beta^T x))}$$

$$\begin{cases} y \rightarrow 1 & \text{if } \beta^T x \rightarrow \infty \\ y = \frac{1}{2} & \text{if } \beta^T x = 0 \\ y \rightarrow 0 & \text{if } \beta^T x \rightarrow -\infty \end{cases}$$

Logistic Regression

- **Logistic Regression (LR)은 대표적인 binary classification 알고리즘**

- positive class에 속할 점수를 $[0, 1]$ 사이로 표현하기 때문에 확률 모형처럼 이용

$$y_{\theta}(x) = \frac{1}{1 + \exp(-\theta^T x)}$$

- 학습 데이터 $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ 가 주어졌을 때, loss function은

$$J(\theta) = - \left[\sum_{i=1}^m y^{(i)} \log h_{\theta}(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_{\theta}(x^{(i)})) \right]$$

Logistic Regression

- **Softmax regression**은 multi class classification 알고리즘으로,
Logistic Regression은 Softmax regression의 특별한 경우
 - Loss function에 대해서는 Word2Vec 때 다룰 예정

$$h_{\theta}(x) = \begin{bmatrix} P(y = 1|x; \theta) \\ \dots \\ P(y = K|x; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^K \exp(\theta^{(j)T} x)} \exp \left(\begin{bmatrix} \theta^{(1)T} x \\ \dots \\ \theta^{(K)T} x \end{bmatrix} \right)$$

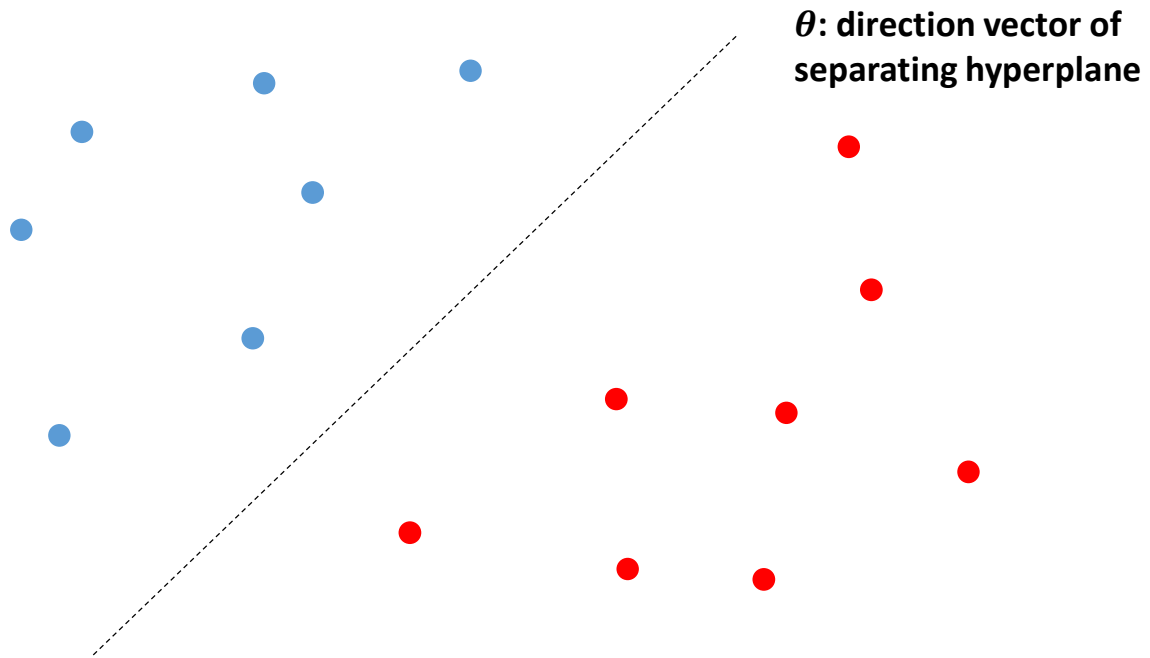
- 학습 데이터 $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ 가 주어졌을 때, loss function은

$$J(\theta) = - \left[\sum_{i=1}^m \sum_{k=1}^K 1\{y^{(i)} = k\} \log \frac{\exp(\theta^{(k)T} x^{(i)})}{\sum_{j=1}^K \exp(\theta^{(j)T} x^{(i)})} \right]$$

Logistic Regression

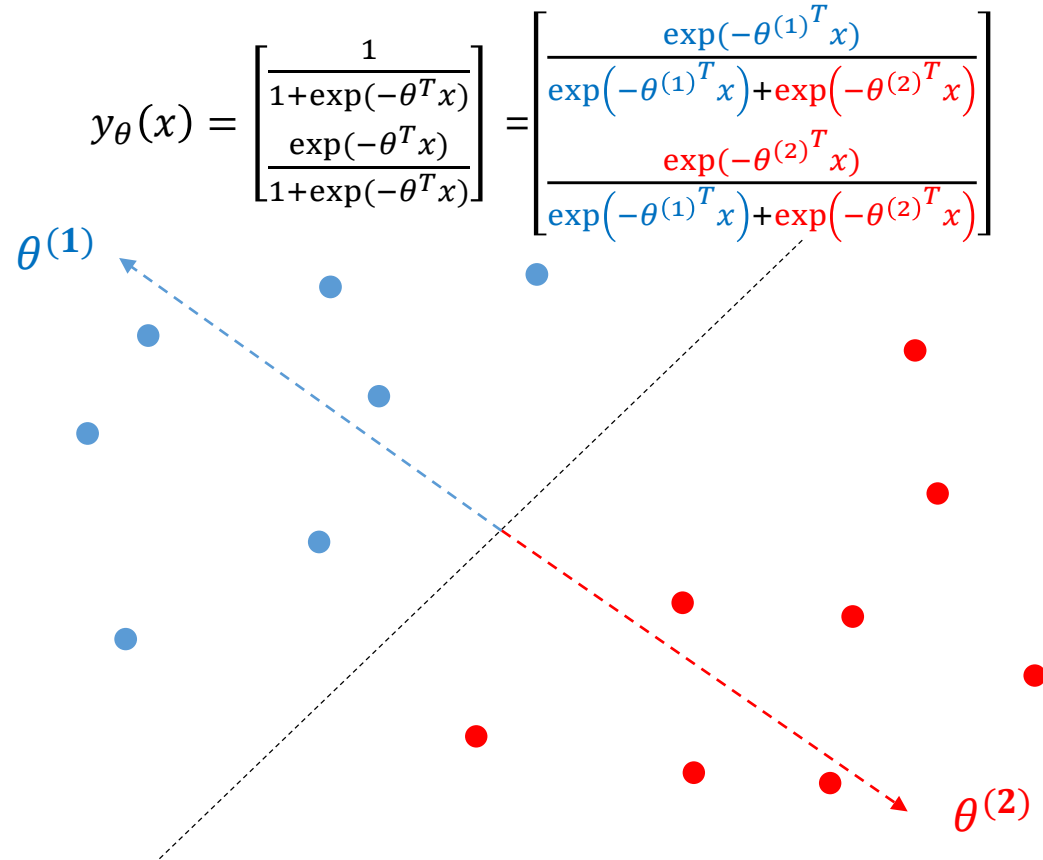
- 일반적으로 LR은 두 클래스의 경계면을 학습한다고 말함

$$y_{\theta}(x) = \frac{1}{1 + \exp(-\theta^T x)}$$



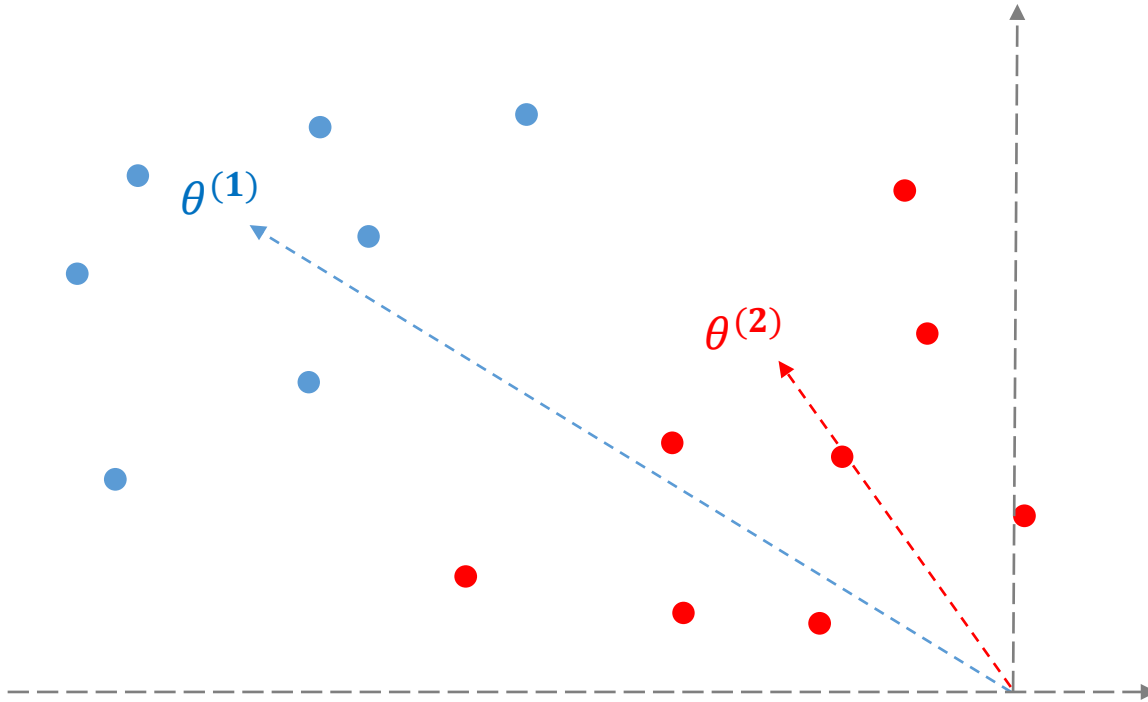
Logistic Regression

- 본래는 Softmax regression 형식으로 학습되며, $\theta = \theta^{(1)} - \theta^{(2)}$ 로 표현된 것
- $\theta^{(i)}$ 는 클래스 i의 대표 벡터로 해석할 수 있음 (Cosine similarity와 유사)



Logistic Regression

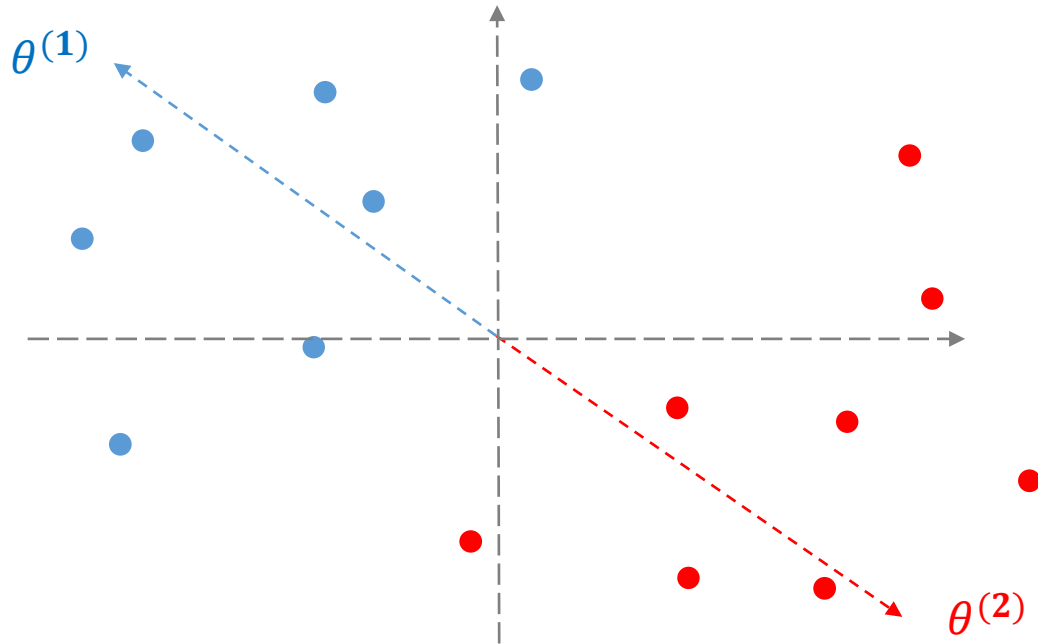
- 두 클래스의 데이터가 같은 방향에 존재할 경우 $\theta^{(i)}$ 만으로는 두 클래스를 구분하기 어려울 수도 있음
 - 단어 빈도 벡터의 값은 모두 양수



Logistic Regression

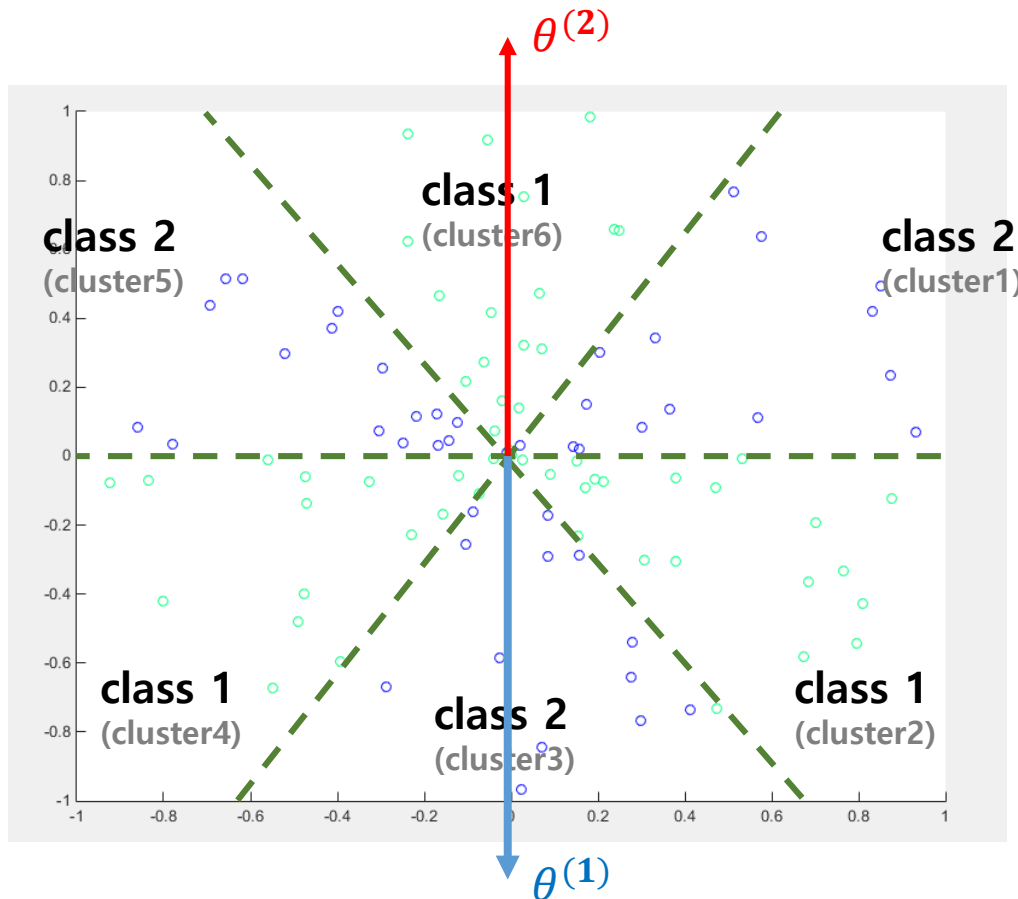
- Bias term은 클래스를 잘 구분하도록 입력변수 x 를 평행이동 시키는 역할

$$\exp(\theta^T x) = \exp(\theta_0 + \theta_1 x_1 + \dots + \theta_p x_p) = \exp(\theta_1(x_1 - k_1) + \dots + \theta_p(x_p - k_p))$$



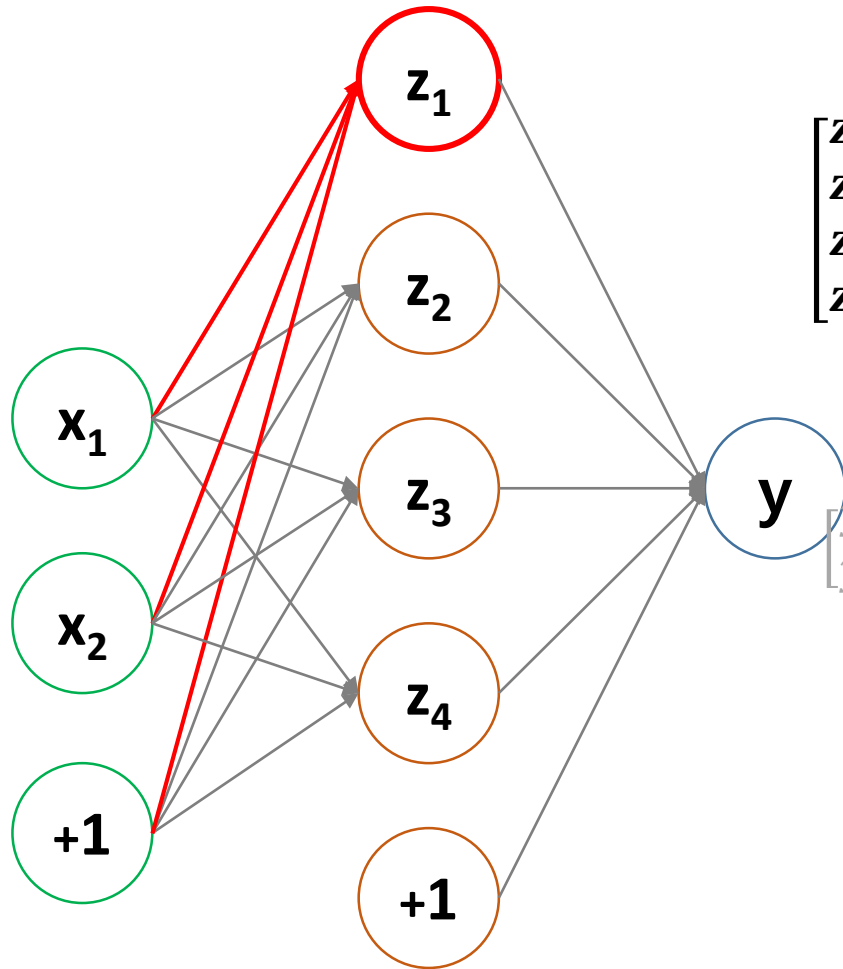
Logistic Regression

- LR은 각 클래스별로 하나의 대표 벡터 $\theta^{(i)}$ 를 학습하기 때문에, 클래스 별로 데이터가 흩어져 있을 경우에는 (linear inseparable) 잘 작동하지 않음



Neural Network

- Sigmoid를 이용하는 feed-forward neural network는 logistic regression을 여러번 하는 것과 동일

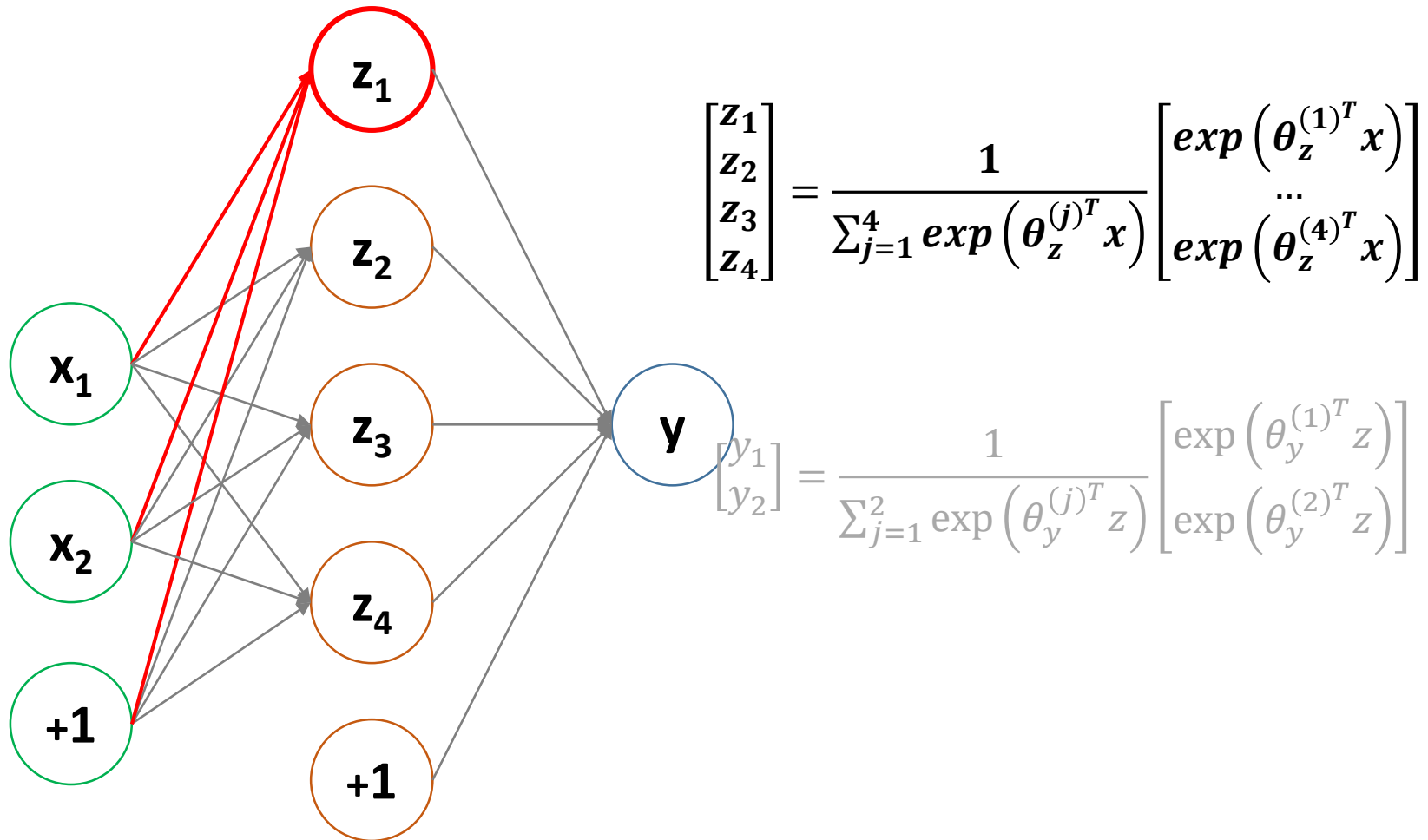


$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \frac{1}{\sum_{j=1}^4 \exp(\theta_z^{(j)T} x)} \begin{bmatrix} \exp(\theta_z^{(1)T} x) \\ \dots \\ \exp(\theta_z^{(4)T} x) \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{1}{\sum_{j=1}^2 \exp(\theta_y^{(j)T} z)} \begin{bmatrix} \exp(\theta_y^{(1)T} z) \\ \exp(\theta_y^{(2)T} z) \end{bmatrix}$$

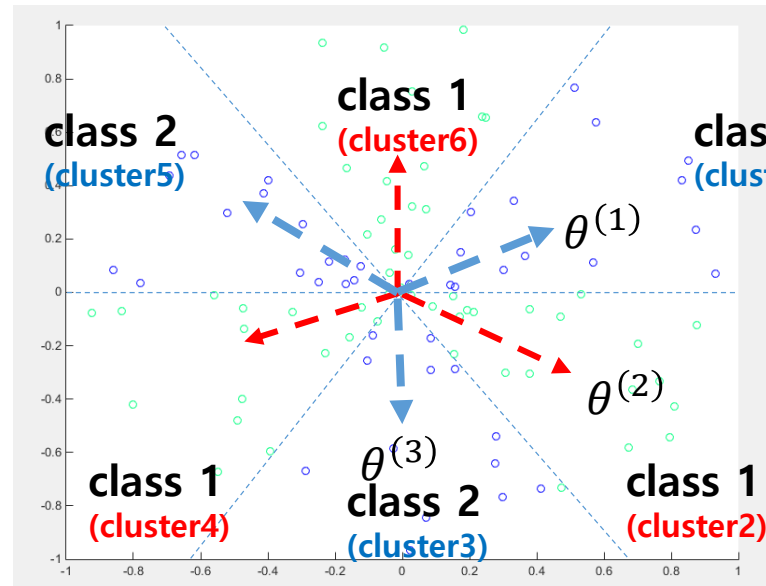
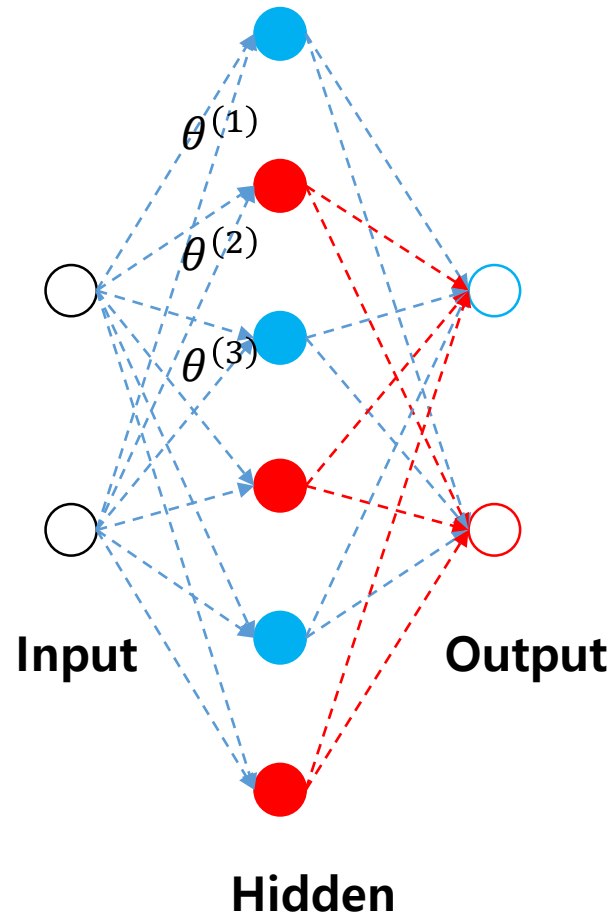
Neural Network

- Sigmoid를 이용하는 feed-forward neural network는 logistic regression을 여러번 하는 것과 동일



Neural Network

- Neural network의 hidden layers 역할은 linear inseparable한 데이터를 조금씩 linear separable한 공간(representation)으로 바꾸는 것



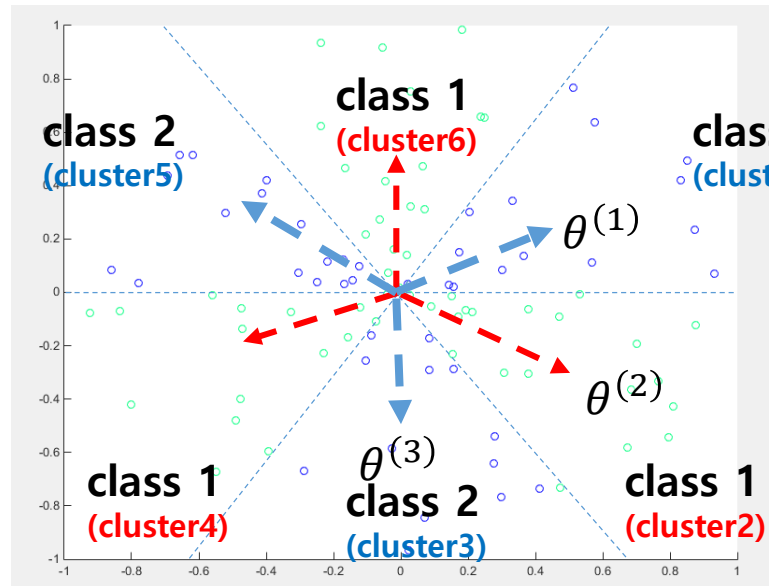
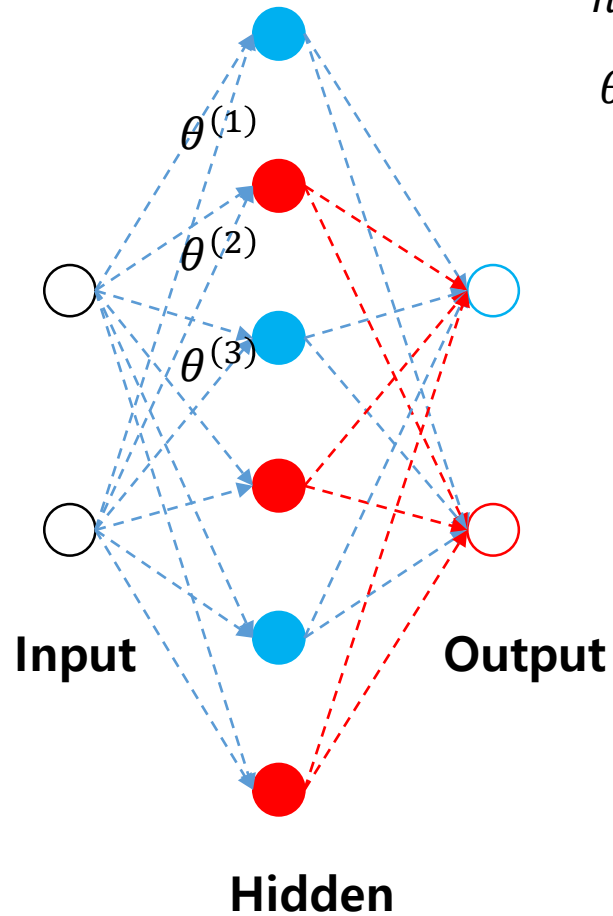
Neural Network

- Sigmoid는 데이터를 Boolean representation에 가깝게 변형시킴

Hidden to output parameters

$$h^{(1)} = [1, \quad 0.3, \quad 0.1, \quad 0.03, \quad 0.1, \quad 0.3]$$

$$\theta^{h \rightarrow o_1} = [1, \quad -1, \quad 1, \quad -1, \quad 1, \quad -1]$$



Regularization

- 모델의 복잡도에 제약/페널티를 부여

- 학습데이터에 과적합 (overfitting)하는 것을 방지
- L2, L1 cost가 대표적
- Logistic Regression 모델의 경우,

$$\text{Loss} = \sum_i^n \left(y_i - \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_{i1} + \dots \beta_p x_{ip}))} \right)^2$$

L1 regularization

$$\text{Loss} = \sum_i^n \left(y_i - \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_{i1} + \dots \beta_p x_{ip}))} \right)^2 + \lambda \sum_j^p |\beta_j|$$

L2 regularization

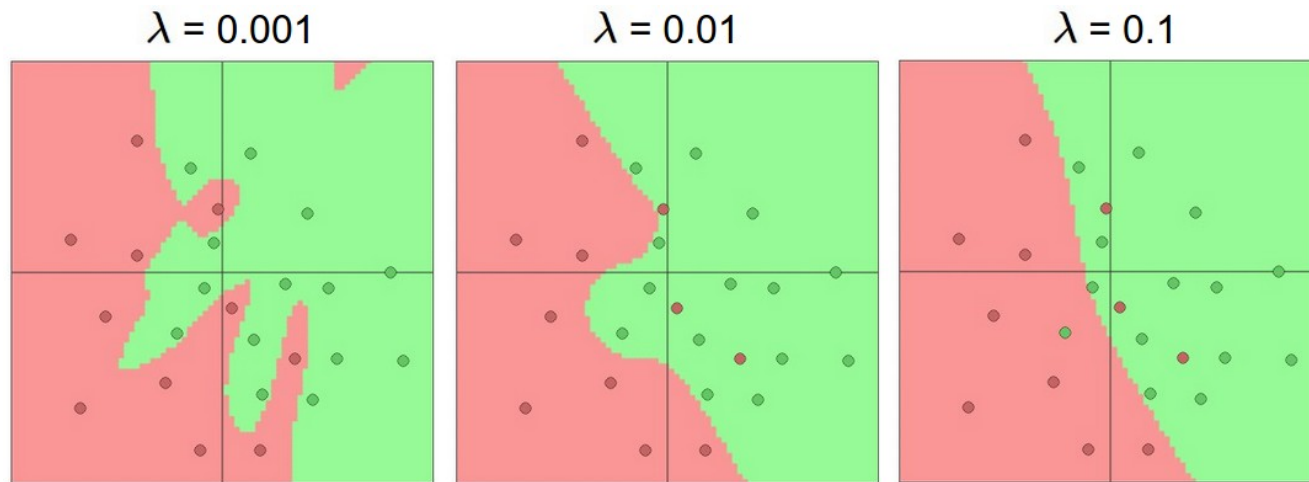
$$\text{Loss} = \sum_i^n \left(y_i - \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_{i1} + \dots \beta_p x_{ip}))} \right)^2 + \lambda \sum_j^p |\beta_j|^2$$

Regularization

- L2 regularization은 경계면을 날카롭지 않게 하는 역할
 - β_i 중에서 크기가 유독 큰 값이 없도록 만들기 때문

Cost = Loss + Regularization term

$$= \sum_i^n \left(y_i - \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}))} \right)^2 + \lambda \sum_j^p |\beta_j|^2$$



Regularization

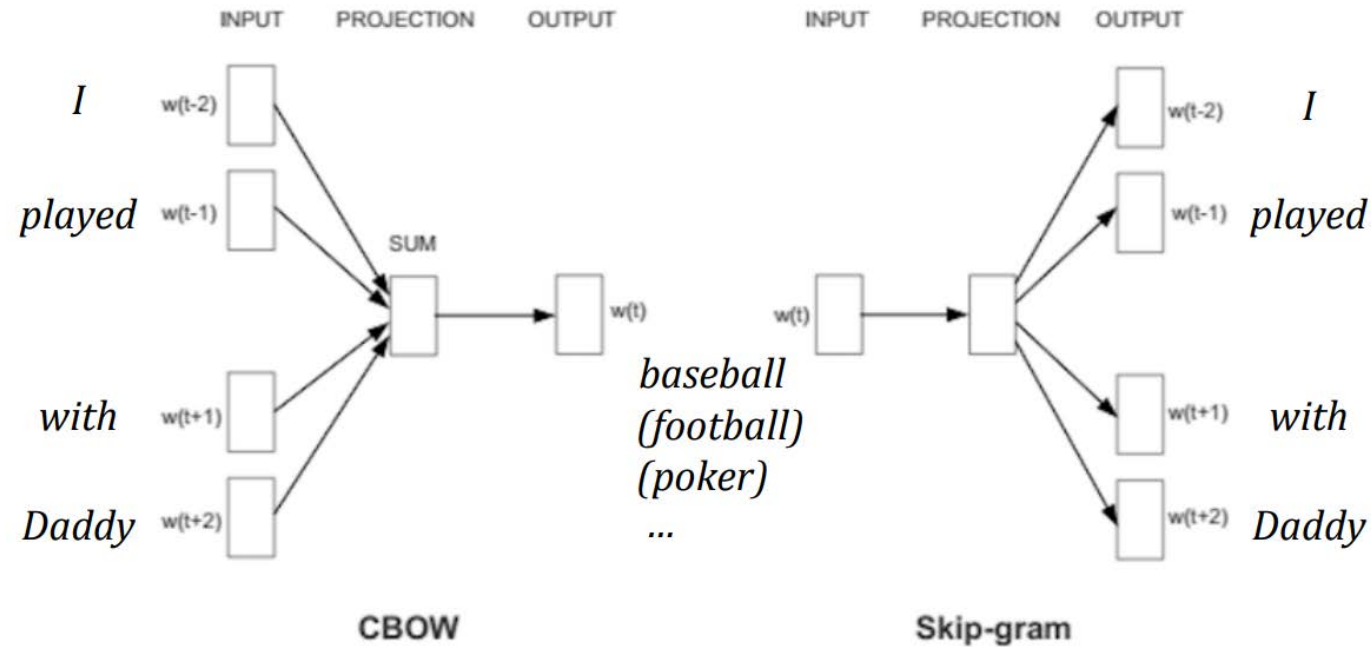
- L1 regularization은 입력변수 계수 β_j 중 일부를 0에 매우 가깝게 만들어, 중요한 입력변수를 선택하는 효과가 있음 (sparse modeling)
 - L1은 L0 regularization의 근사로, L0이 미분이 되지 않기 때문에 쓰이게 되었으나 중요한 변수를 고르는 효과 측면에서는 비슷

$$\text{Cost} = \sum_i^n \left(y_i - \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}))} \right)^2 + \lambda \sum_j^p |\beta_j|$$

Word2vec / Doc2vec

의미 기반 단어 임베딩 (Document representation) – Word2vec

- “친구를 보면 그 사람을 알 수 있다” (A man is known by the company he keeps)
 - 분산가설: 단어의 의미를 그 주위의 단어 분포로부터 유추할 수 있다
 - 유사한 ‘친구’를 가진 단어들을 유사한 벡터로 학습
 - $\text{baseball} = \langle 0.7, 0.1, \dots, 0.3, 0.0 \rangle$, $\text{football} = \langle \dots \rangle$



Word2Vec

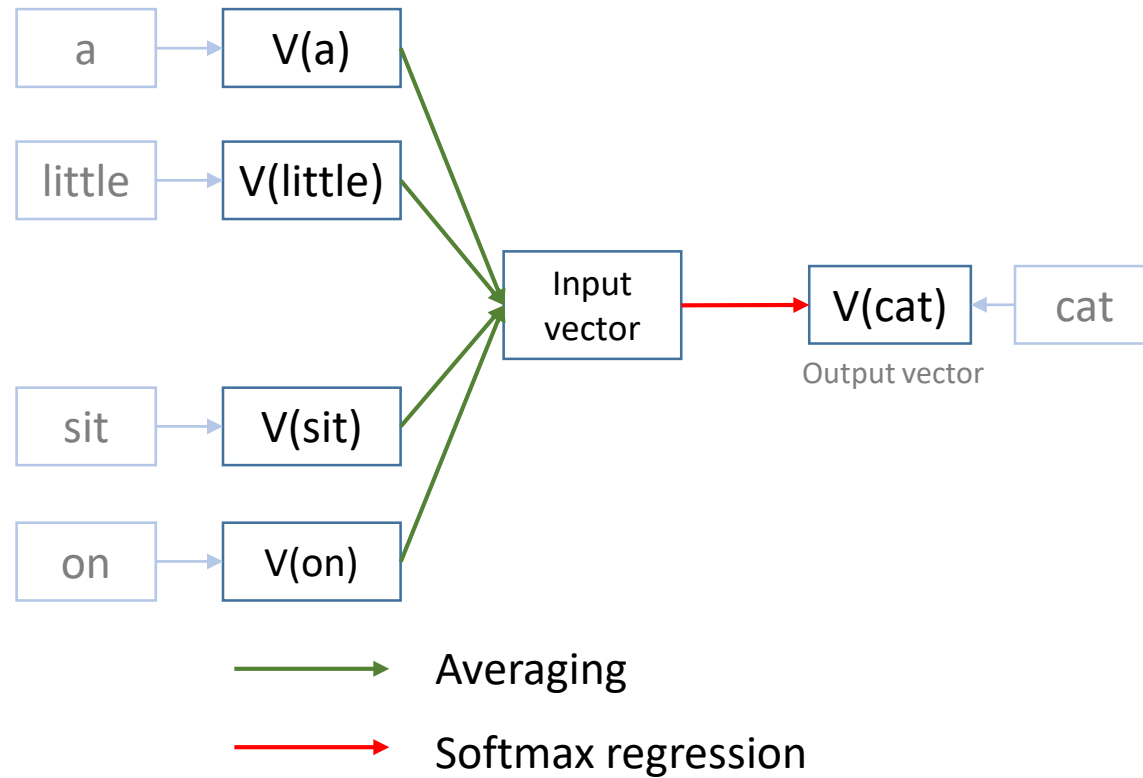
- Word2Vec은 $|V|$ class Softmax regression을 이용하여 각 단어의 벡터를 학습시키는 classifier

Lookup table

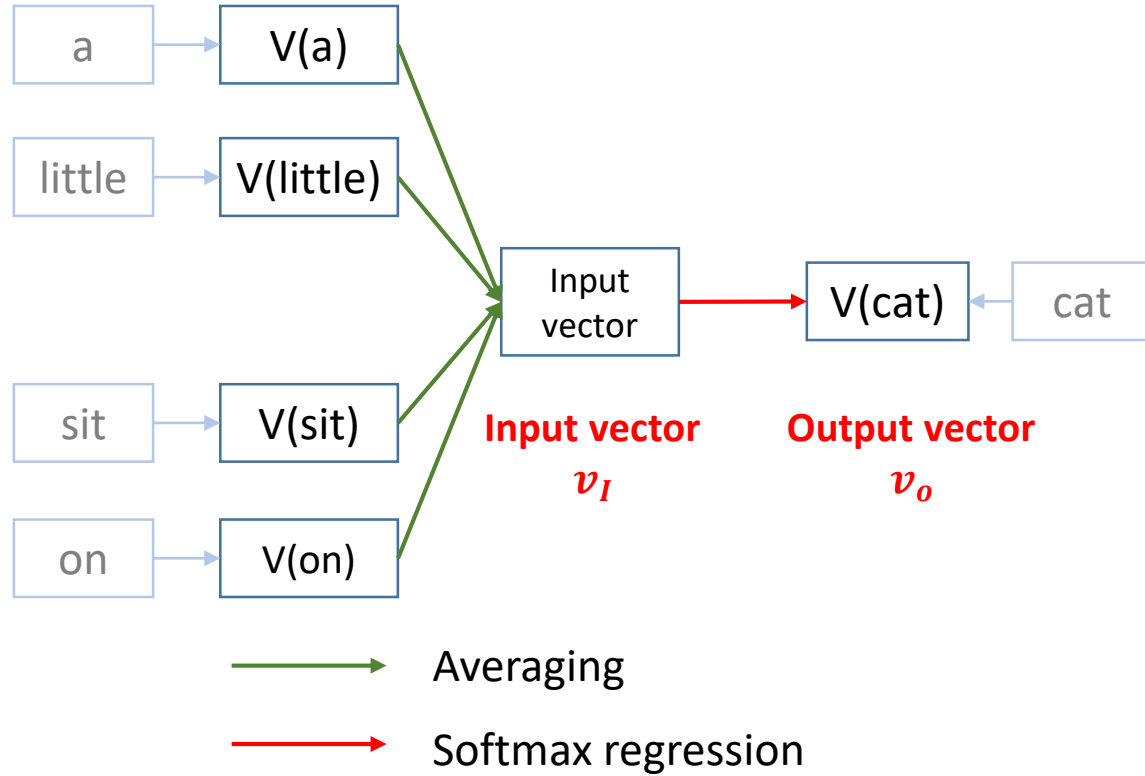
```
{cat: [.31, -.22, .54]  
dog: [.22, -0.3, -0.52]  
on: ...  
the: ...  
little: ...  
... }
```

Lookup word vector

$V(\text{'cat'}) = [.31, -.22, .54]$

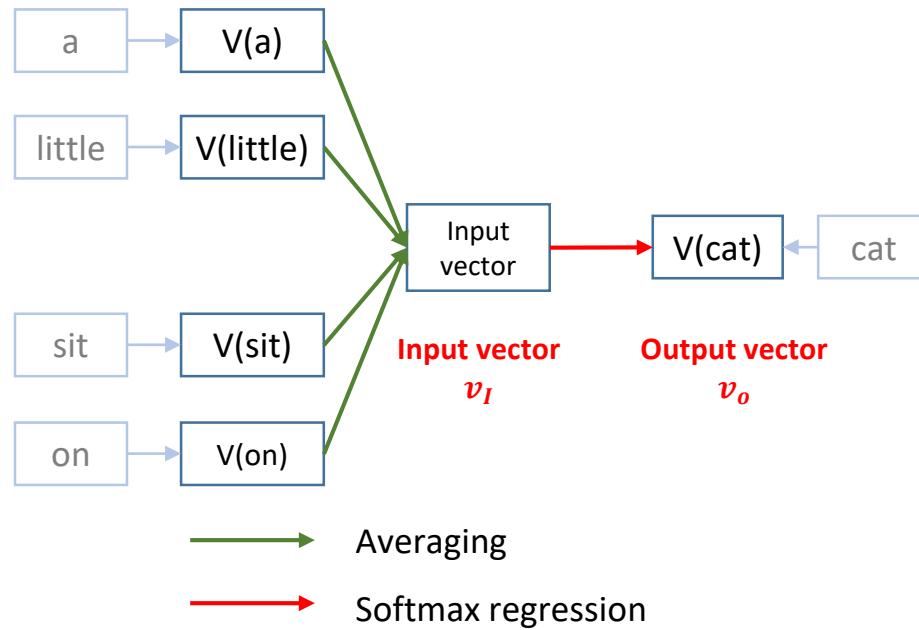


Word2Vec



$$y_j = \frac{\exp(v_I^T v_{O_j})}{\sum_k \exp(v_I^T v_{O_k})}$$

Word2Vec



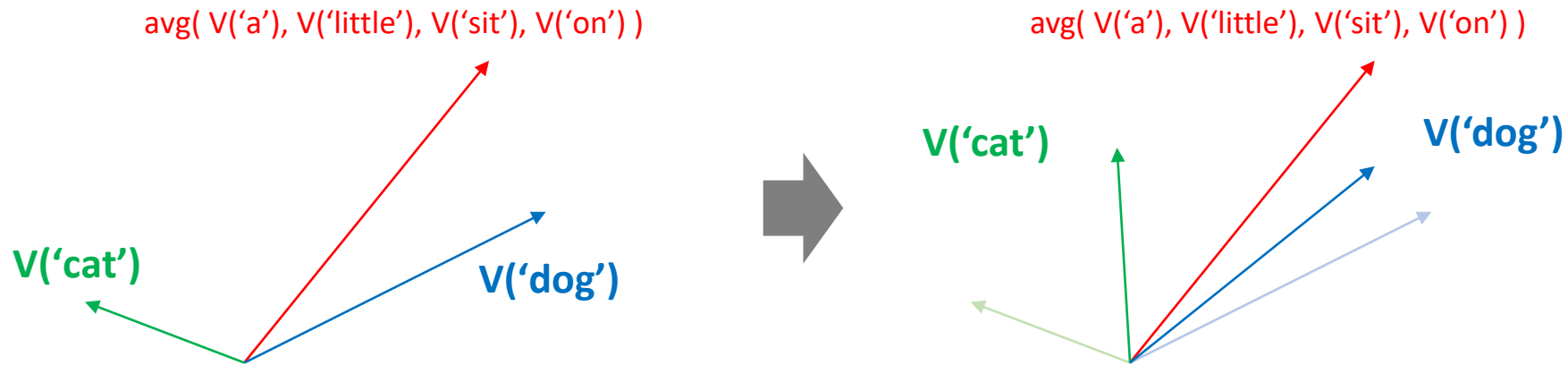
$$h_{\theta}(x) = \begin{bmatrix} P(w_{(t)} = \text{cat}) \\ P(w_{(t)} = \text{dog}) \\ P(w_{(t)} = \text{table}) \\ \dots \\ P(w_{(t)} = \text{Vocab}) \end{bmatrix} = \frac{1}{\sum_{j=1}^{|V|} \exp(\theta^{(j)T} x)} \begin{bmatrix} \exp(v(\text{cat})^T v_I) \\ \exp(v(\text{dog})^T v_I) \\ \exp(v(\text{table})^T v_I) \\ \dots \\ \exp(v(\text{Vocab})^T v_I) \end{bmatrix}$$

x : Input vector (average of contextual word vector)

Word2Vec

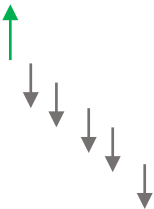
- 'cat'과 'dog'이 비슷한 word embedding vector를 가지는 것은, 두 단어가 비슷한 문맥(contextual word distribution)을 가지기 때문이다
 - 'cat', 'dog'의 두 단어 벡터 모두 context vector에 가깝게 이동
 - Context vector와 차이가 많이나는 단어 'cat'은 학습량 (벡터의 변화량)도 큼

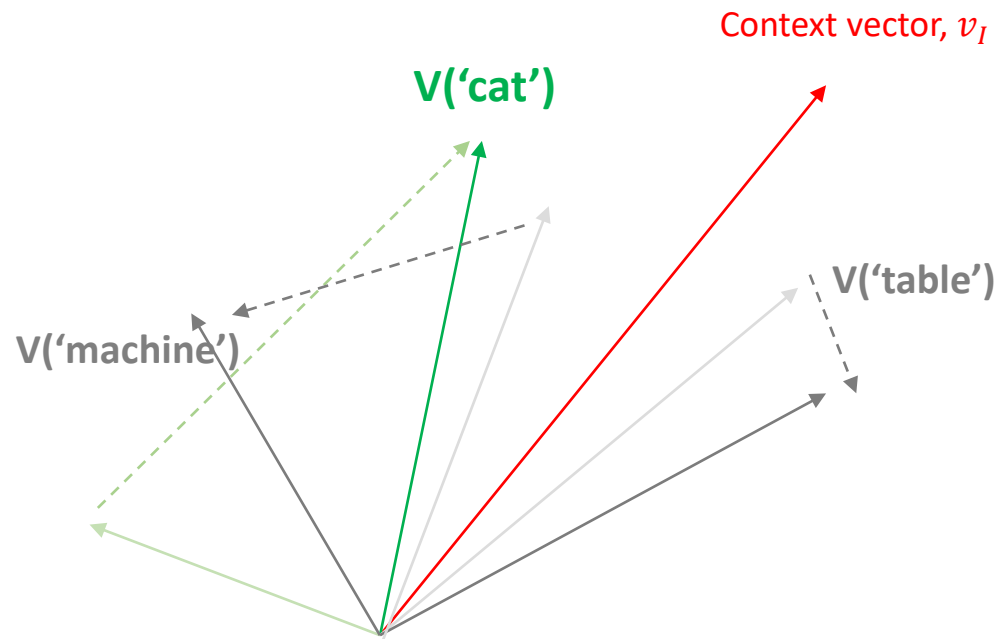
Softmax regression loss



Word2Vec

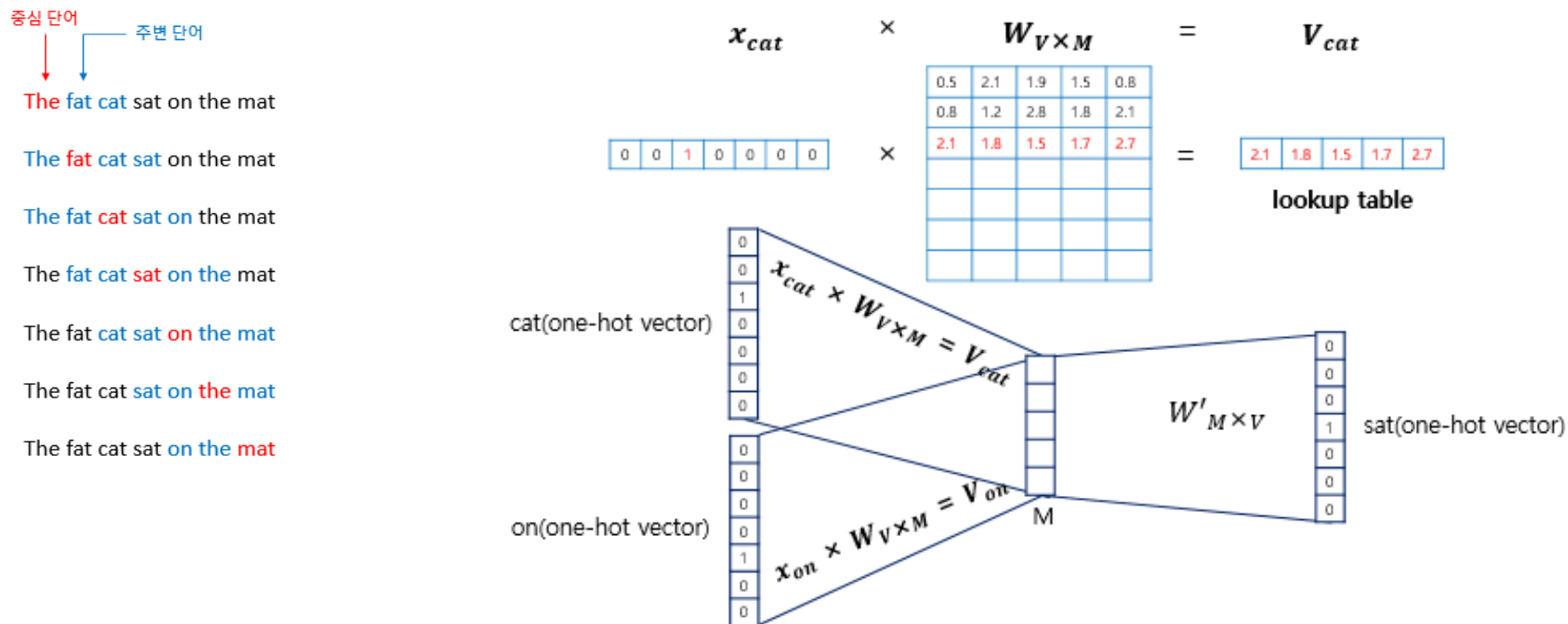
- Softmax regression은 알맞은 단어는 context vector 방향으로 당기고, 틀린 단어는 반대 방향으로 밀어내는 역할

$$h_{\theta}(x) = \begin{bmatrix} P(w_{(t)} = cat) \\ P(w_{(t)} = dog) \\ P(w_{(t)} = table) \\ \dots \\ P(w_{(t)} = Vocab) \end{bmatrix} = \frac{1}{\sum_{j=1}^{|V|} \exp(\theta^{(j)T} x)} \begin{bmatrix} \exp(v(cat)^T v_I) \\ \exp(v(dog)^T v_I) \\ \exp(v(table)^T v_I) \\ \dots \\ \exp(v(Vocab)^T v_I) \end{bmatrix}$$




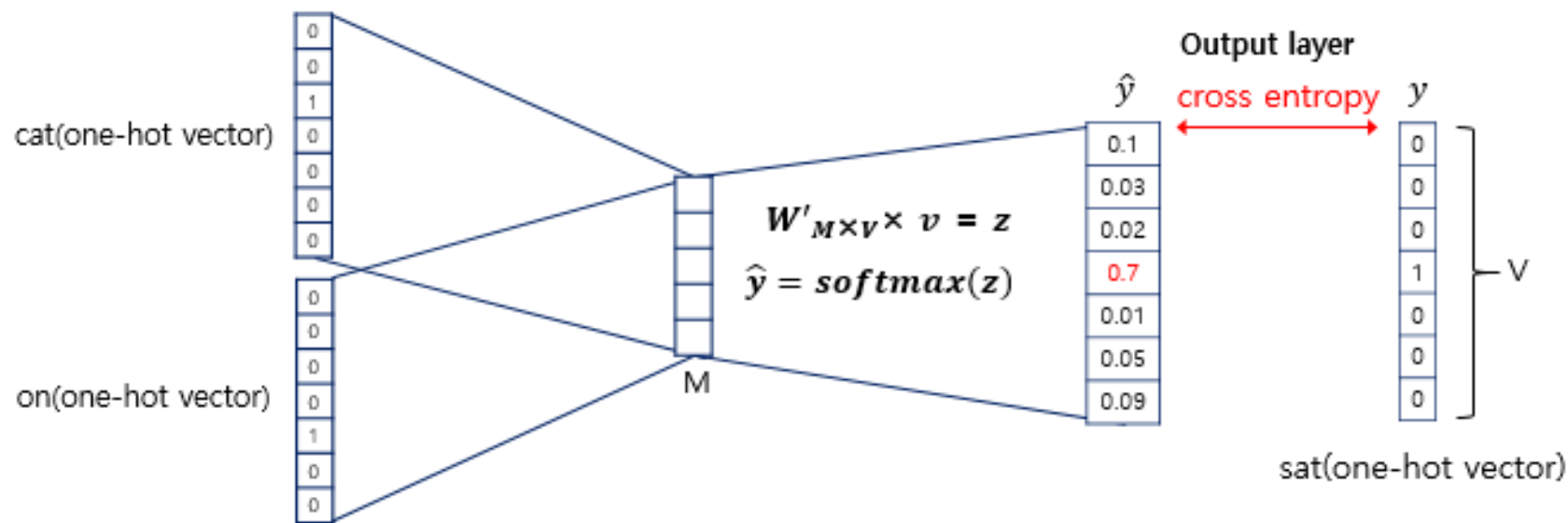
Word2Vec

- 구체적으로 도식화하면 아래와 같은 w 매트릭스를 업데이트 하는 개념



Word2Vec

- 구체적으로 도식화하면 아래와 같은 매트릭스를 업데이트 하는 개념



Word2Vec

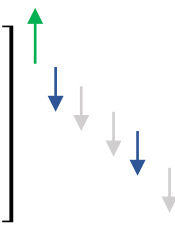
- Softmax regression은 한 단어의 벡터를 학습하기 위해 v 개의 모든 단어의 벡터를 수정하기 때문에 계산량이 매우 큼

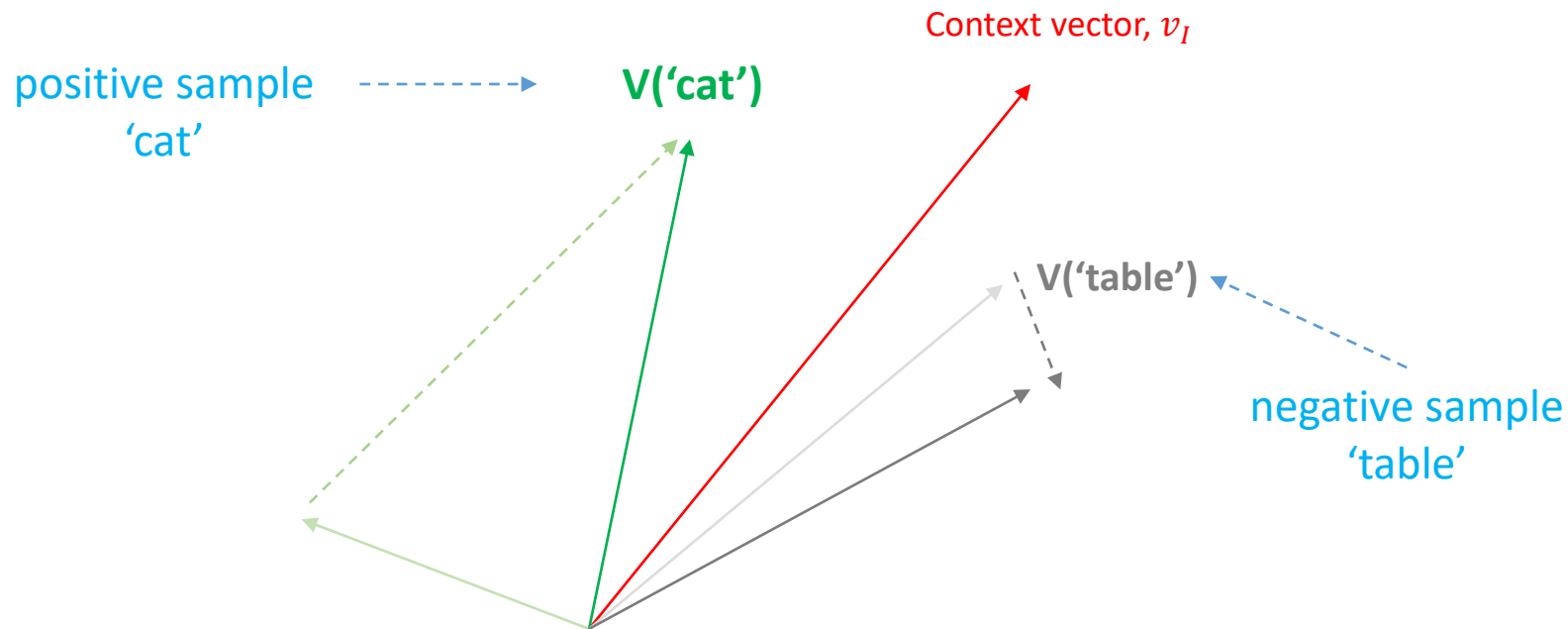
◦ 학습 데이터 $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ 가 주어졌을 때, loss function은

$$J(\theta) = - \left[\sum_{i=1}^m \sum_{k=1}^V 1\{w_{(t)} = cat\} \log \frac{\exp(\theta^{(j)T} x^{(i)})}{\sum_{j=1}^K \exp(\theta^{(j)T} x^{(i)})} \right]$$

Word2Vec

- Negative sampling은 cat이 아닌 모든 단어를 밀어내는 것이 아니라, 임의로 샘플링한 단어들에 대해서만 context vector에서 밀어내는 것

$$h_{\theta}(x) = \begin{bmatrix} P(w_{(t)} = cat) \\ P(w_{(t)} = dog) \\ P(w_{(t)} = table) \\ \dots \\ P(w_{(t)} = Vocab) \end{bmatrix} = \frac{1}{\sum_{j=1}^{|V|} \exp(\theta^{(j)T} x)} \begin{bmatrix} \exp(v(cat)^T v_I) \\ \exp(v(dog)^T v_I) \\ \exp(v(table)^T v_I) \\ \dots \\ \exp(v(Vocab)^T v_I) \end{bmatrix}$$


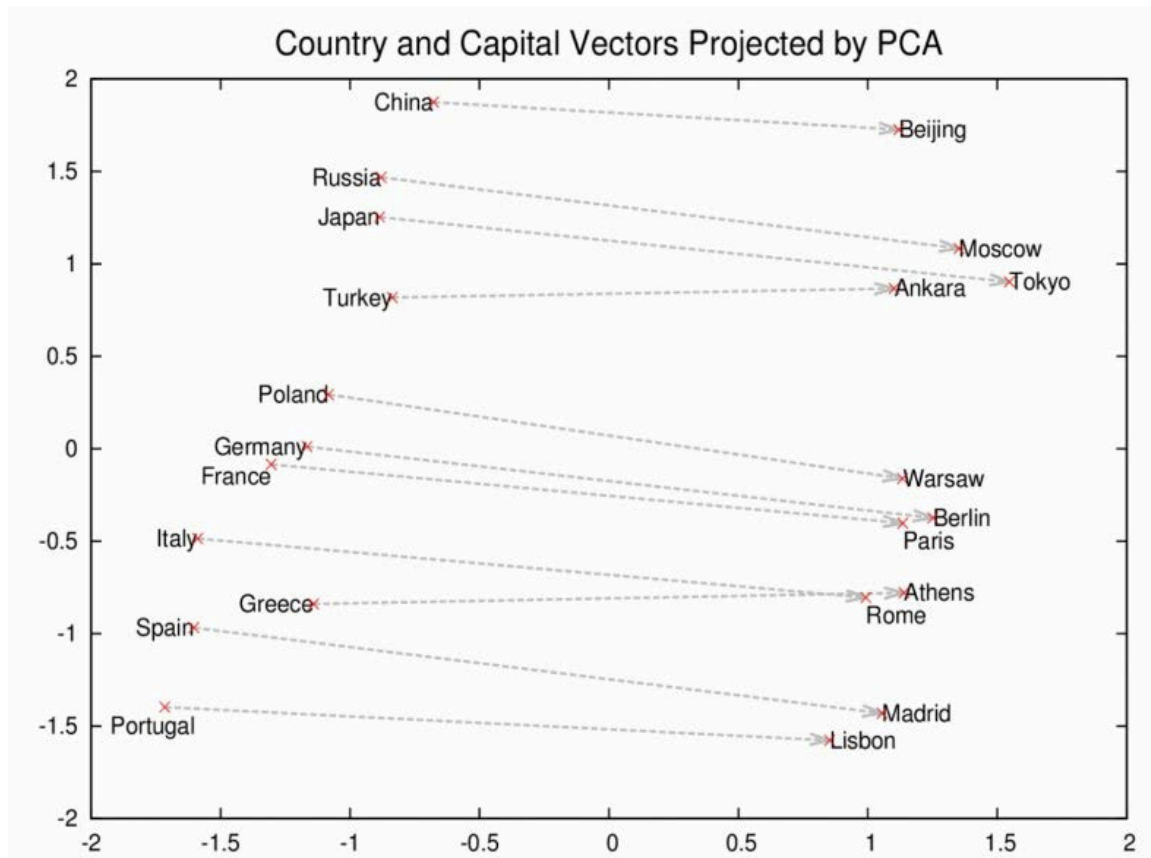


Word2Vec

- ‘cat’이 등장한 많은 문장이 있기 때문에, 한 번에 조금씩 $v(\text{'cat'})$ 을 학습하면 여러 문맥들을 모두 고려할 수 있음.
 - [a, little, cat, sit, on, table],
[my, pretty, cat, ran, out],
....
 - 조금씩 학습한다 = 작은 learning rate를 이용한다

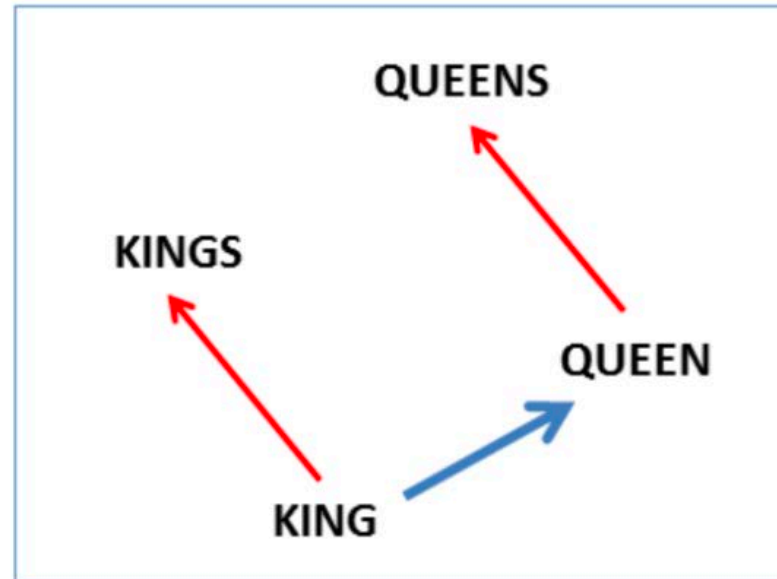
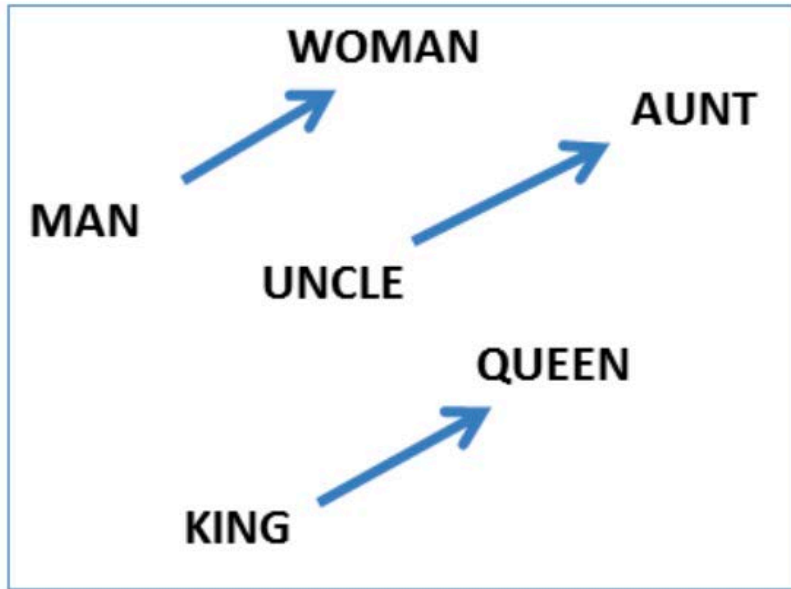
Word2Vec

- Word2Vec은 단어간의 관계성이 추출되는 효과가 있음
 - “ $v(\text{나라}) - v(\text{수도})$ ” 벡터가 나라마다 비슷



Word2Vec

- Word2Vec은 단어간의 관계성이 추출되는 효과가 있음
 - $V(\text{kings}) - V(\text{king}) = V(\text{queens}) - V(\text{queen})$



Word2Vec

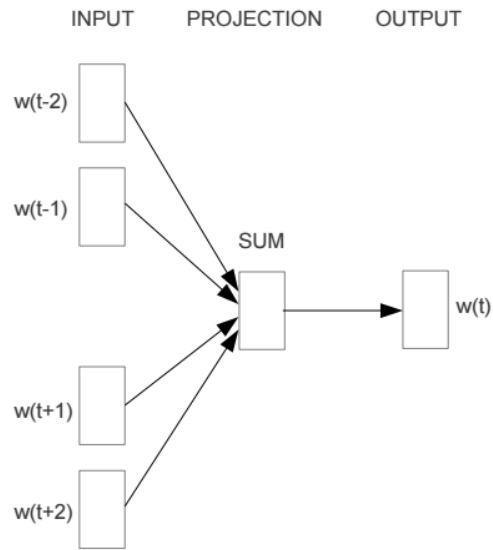
- Neural network language modeling (NNLM) 은 2003년 제안된 개념 (Bengio et al., 2003).
- v 복잡한 구조로 인해 억 단위의 단어에 대한 계산이 어려움
- v 이러한 단점을 보완하고자 간단하면서도 우수한 성능을 보이는
- Word2Vec 모델이 제안됨 (Mikolov et al., 2013)

“박원순” 유사			“사드” 유사		
	word	cosine		word	cosine
0	서울시장	0.478807	0	배치	0.504133
1	오세훈	0.457326	1	성주	0.503673
2	구의역	0.439984	2	한반도	0.439496
3	좌편향	0.410353	3	미사일방어	0.438899
4	이재명	0.382454	4	중국	0.433367
5	정몽준	0.361279	5	한미	0.433314
6	민평련	0.359114	6	북핵	0.420724
7	충남지사	0.358133	7	미사일	0.404025
8	안희정	0.357108	8	국익	0.393909
9	시장님	0.352705	9	방어	0.39005
10	서울시	0.344189	10	한민구	0.389631
11	권선택	0.337724	11	안보	0.386246
12	사단법인	0.330932	12	설득	0.385805
13	시장	0.325118	13	국방부	0.382189
14	허드	0.323974	14	싸드	0.374187

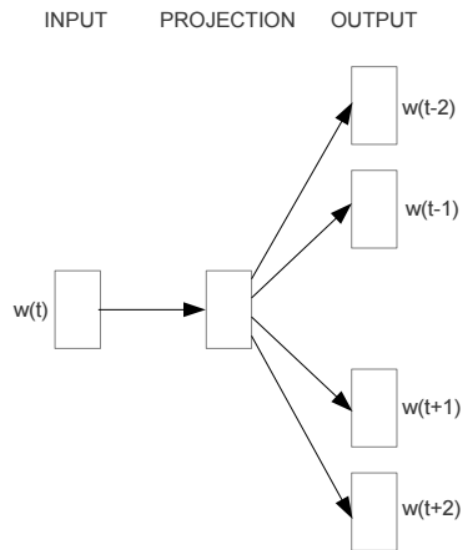
Corpus : 2016년 6월~8월15일 정치분야 뉴스

Word2Vec

- Word2Vec은 CBOW, Skip-gram 두가지 형태가 있음
 - CBOW는 주위 단어로 현재 단어 $w(t)$ 를 예측하는 모델
 - Skipgram은 현재 단어 $w(t)$ 로 주위 단어 모두를 예측하는 모델



CBOW

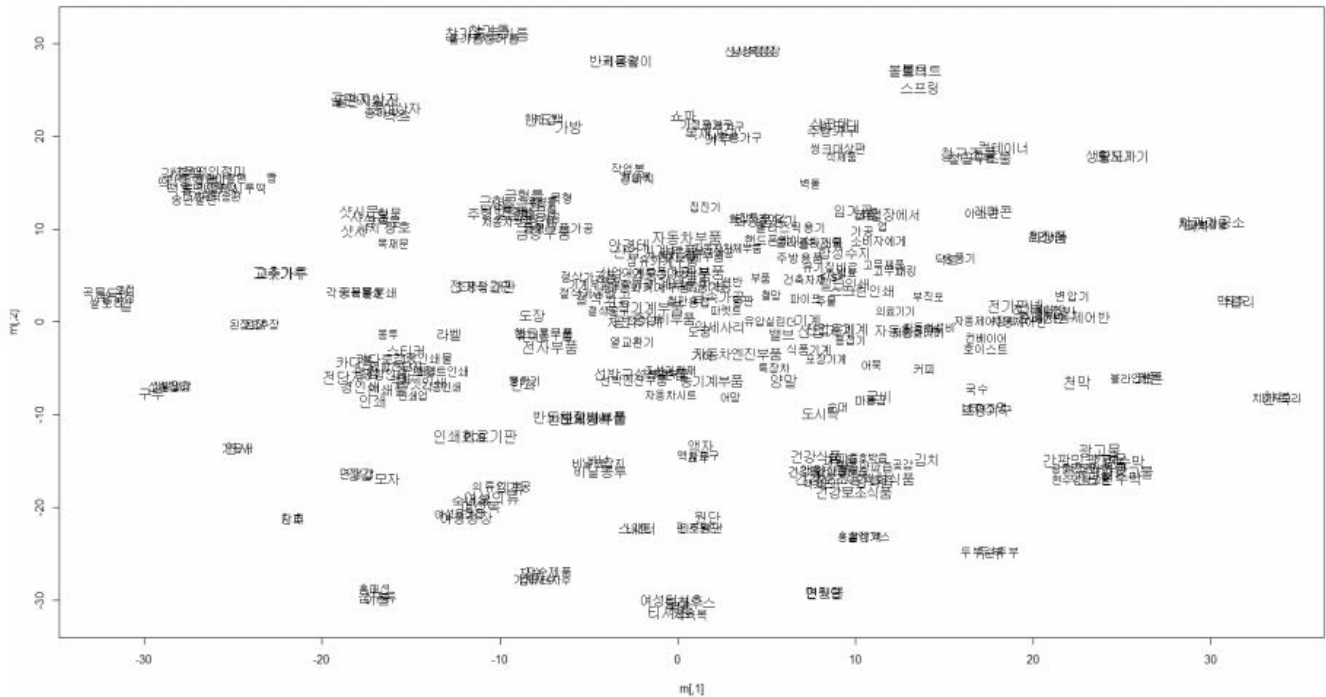


Skip-gram

Word2vec 예시

- Word2vec 활용 정도
 - http://www.dbpia.co.kr/search/topSearch?startCount=0&collection=ALL&range=A&searchField=ALL&sort=RANK&query=word2vec&srchOption=*&includeAr=false&searchOption=*

- Word2Vec을 활용한 제품군별 시장규모 추정 방법에 관한 연구



〈Figure 2〉 Visualization of Word2Vec training result using t-SNE

KSIC Index	The extracted product names	Cosine Similarity
그림액자표구제품	그림액자표구제품	1.00000
그림액자표구제품	축자	0.97558
그림액자표구제품	표구사	0.97350
그림액자표구제품	표구처리	0.97186
그림액자표구제품	병풍	0.96755
그림액자표구제품	동양화서양화	0.96408
그림액자표구제품	액자주문	0.96396
그림액자표구제품	액자표구등	0.96334
그림액자표구제품	액자틀표구	0.96321
그림액자표구제품	병풍및표구	0.96279
그림액자표구제품	액자표구소매	0.96250
그림액자표구제품	POP액자아크릴상자스카시등	0.96225
그림액자표구제품	문구류전사인쇄	0.96144
그림액자표구제품	액자표구현황판	0.96057
그림액자표구제품	서예표구	0.96040
그림액자표구제품	전사	0.96034
그림액자표구제품	케이스마케팅용품(학용품)	0.96014
그림액자표구제품	플라스틱표면인쇄	0.95989
그림액자표구제품	서예및그림표구	0.95988
그림액자표구제품	축자무역업	0.95970
커튼	커튼	1.00000
커튼	커텐	0.96158
커튼	커튼블라인드	0.94331
커튼	커텐블라인드	0.94096
커튼	커텐브라인드	0.93779
커튼	커텐주문	0.92920
커튼	브라인드커텐	0.91817

Word2vec 예시

- Word2Vec 기반 데이터 시각화를 활용한 건설 재해관련 공공데이터 텍스트 분석

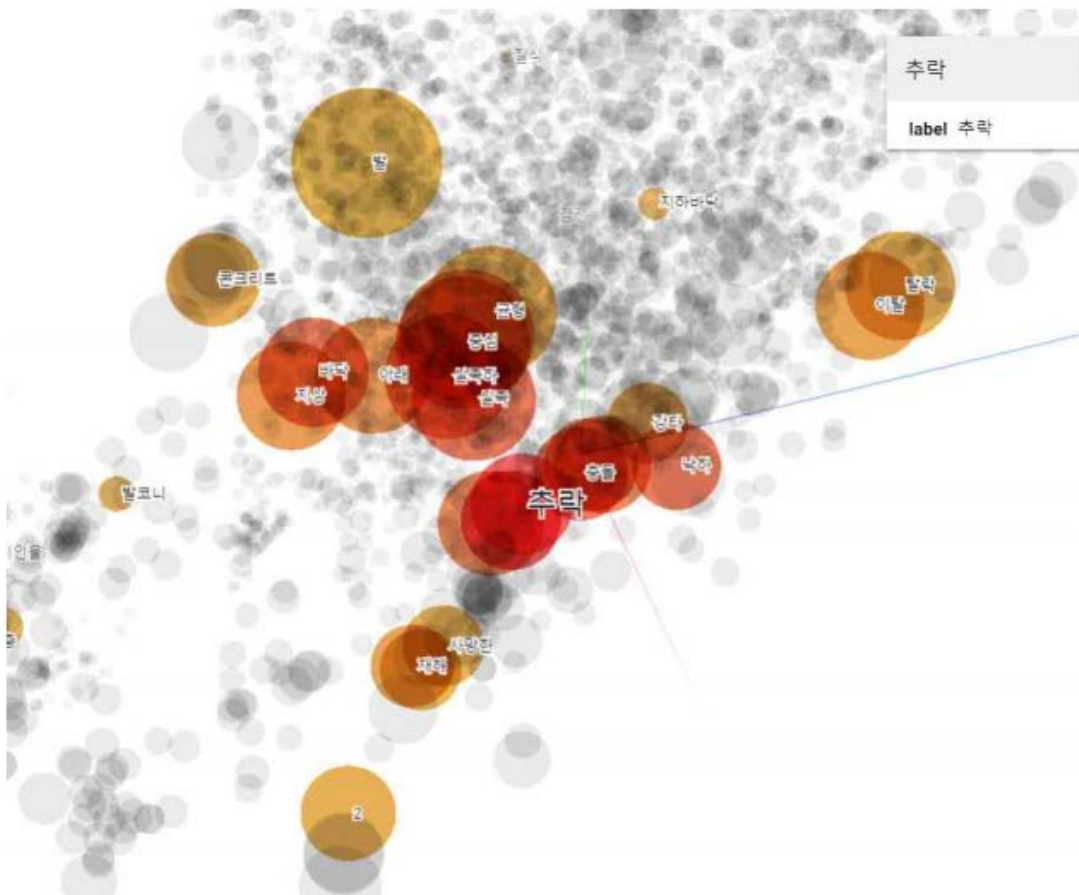


그림 3 Word2Vec t-SNE 시각화 맵 - '추락' 연관어

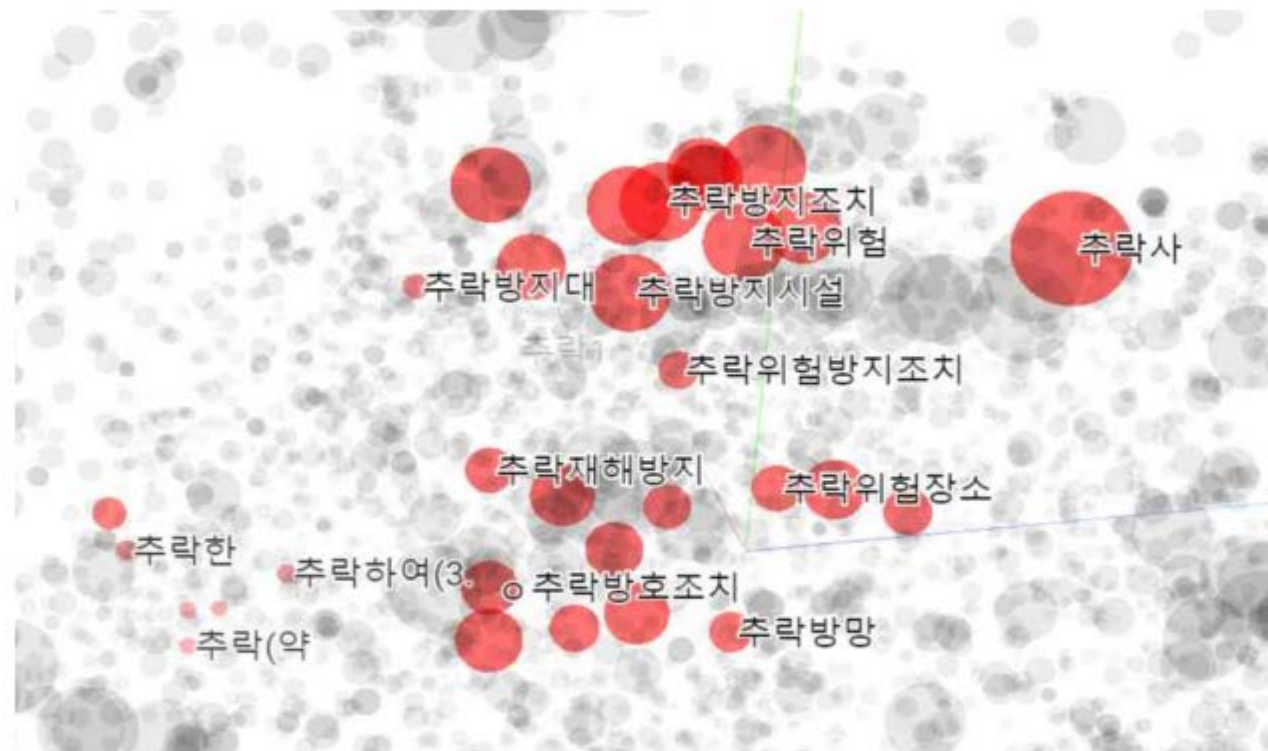
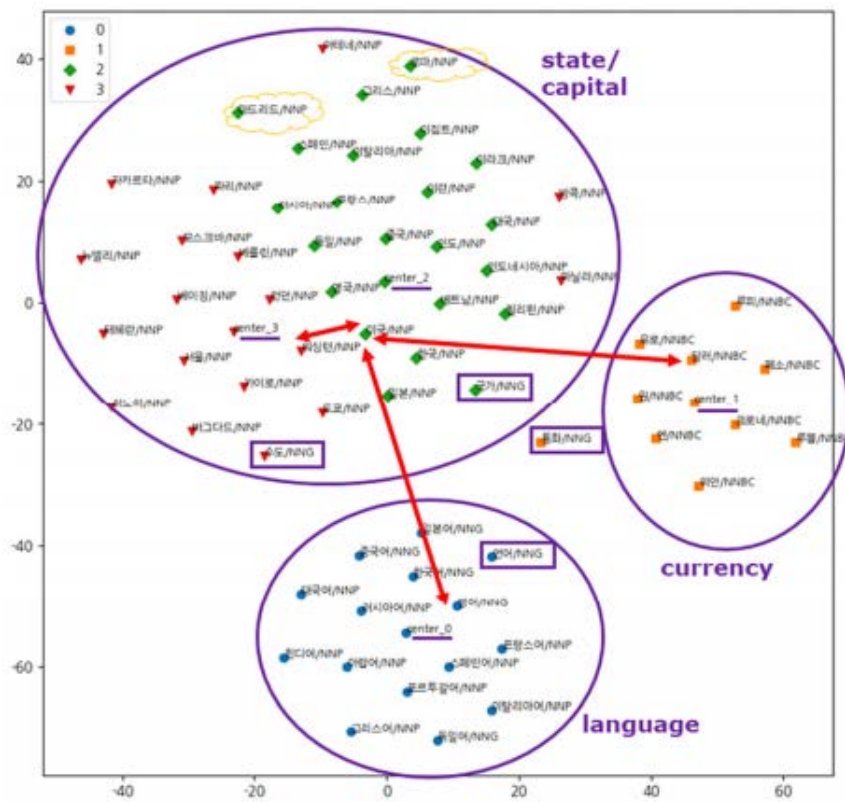


그림 4 Word2Vec t-SNE 시각화 맵 - '추락' 유의어

Word2vec 예시

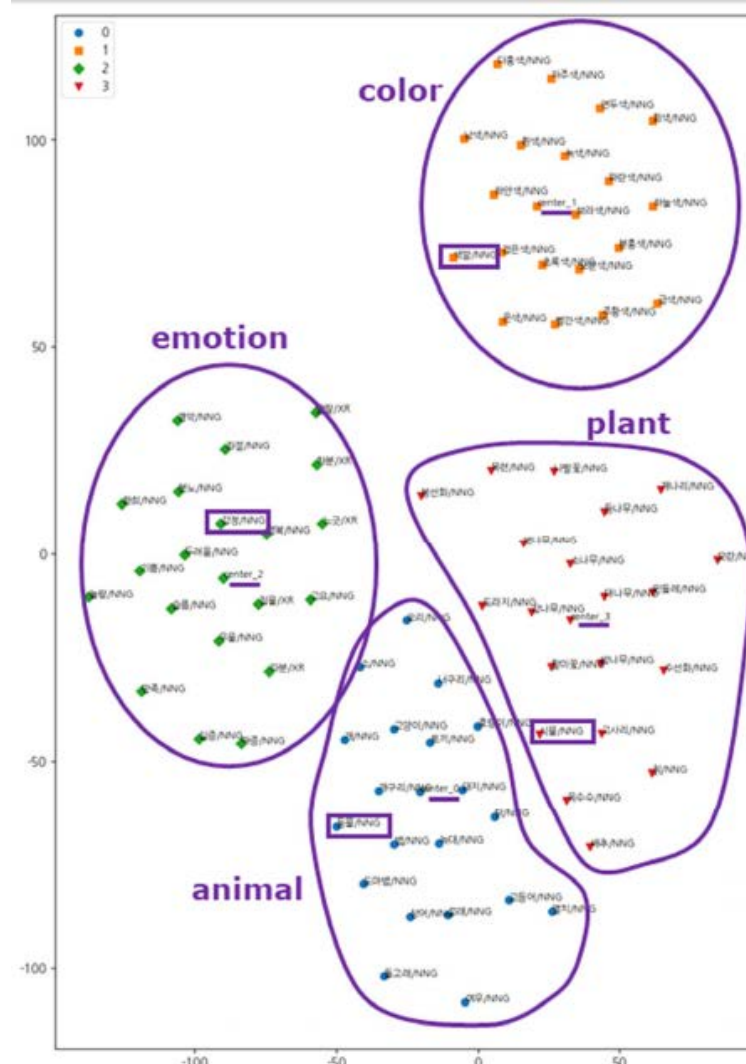
- Word2vec 모델로 학습된 단어 벡터의 의미 관계 분석



(b) t-SNE scatter plot

그림 1 제1 그룹 - 국가/수도/언어/통화

Fig. 1 Group 1 - State/Capital/Language/Currency



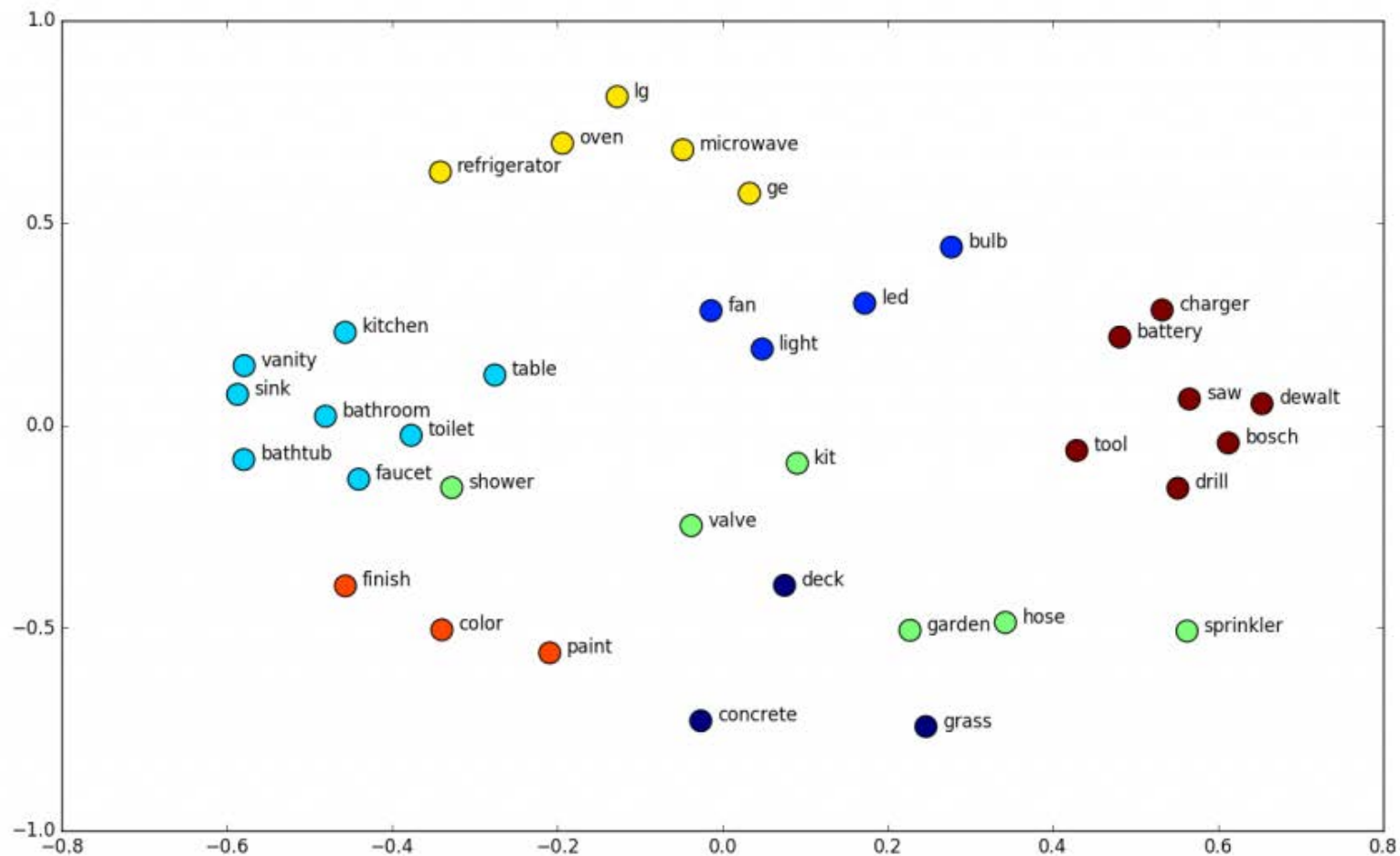
(b) t-SNE scatter plot

그림 3 제3 그룹 - 감정/색깔/동물/식물

Fig. 3 Group 3 - Emotion/Color/Animal/Plant

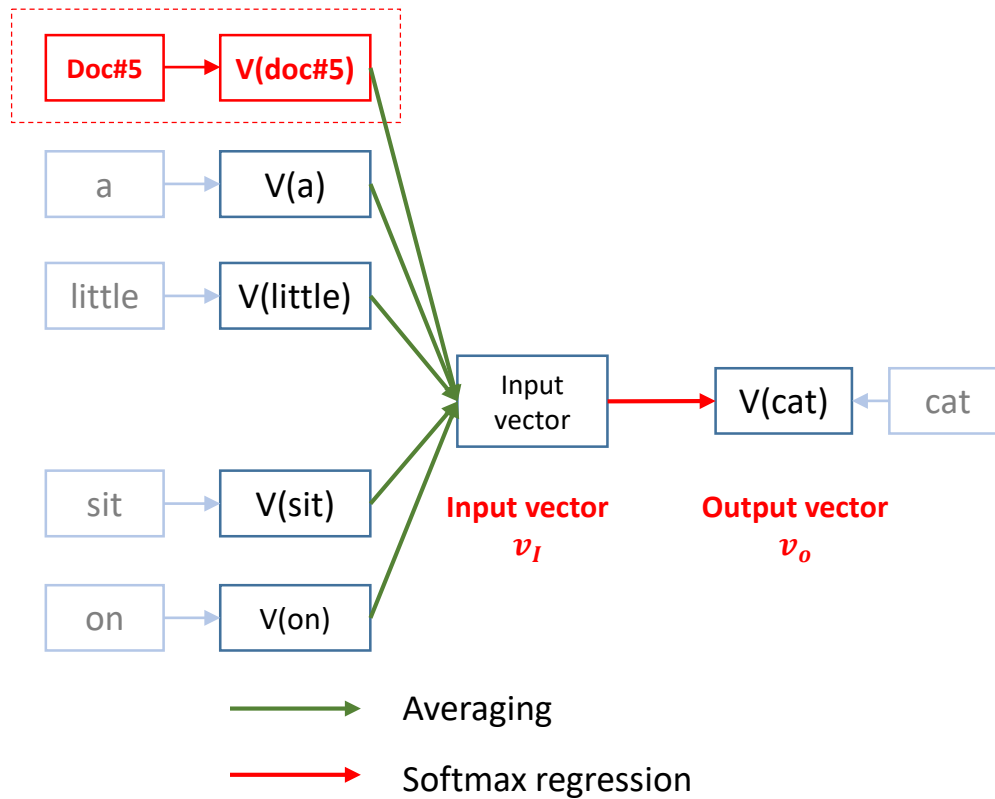
Word2vec 연습 문제

- 아래 word2vec 예시에서 'bathroom'의 임베딩 좌표를 구하고, 가장 가까운 단어 3개를 파악해보세요.



Doc2Vec

- Doc2Vec은 가상의 단어 Document Id를 삽입한 뒤, Word2Vec을 학습하는 것
 - Document id와 word가 같은 embedding 공간에 위치

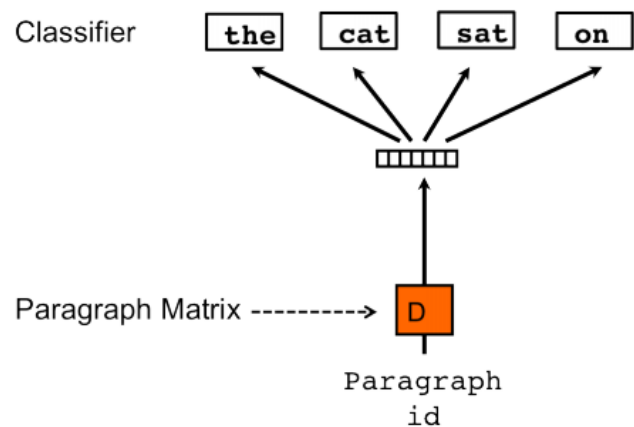


Doc2Vec

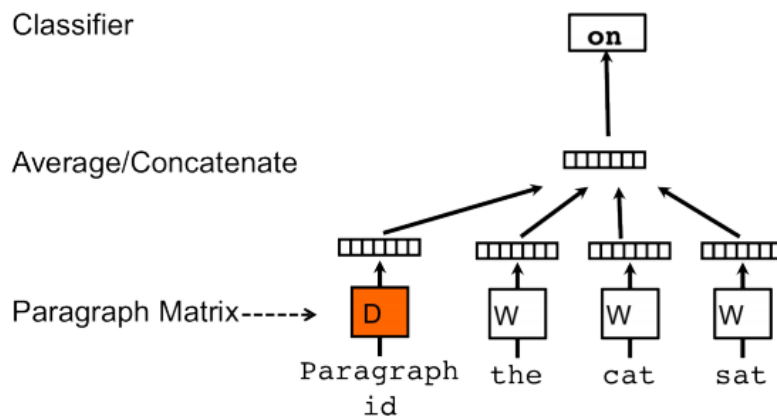
- **문장 혹은 문서에 대해서도 continuous representations으로 학습하는 unsupervised 방법**
 - Distributed memory model of paragraph vector (PV-DM)
 - Distributed bag of words version of paragraph vector (PV-DBOW)
 - Word2vec의 장점을 그대로 가져감.
 - Bag of words에 비해 단어 순서를 고려하고 low-dimensional vector를 생성한다는 장점

Doc2Vec

- Doc2Vec은 가상의 단어 Document Id를 삽입한 뒤, Word2Vec을 학습하는 것
 - Word2Vec처럼 2가지 모델 버전이 있음



Document id로 모든 단어를 예측하는
Softmax regression을 학습

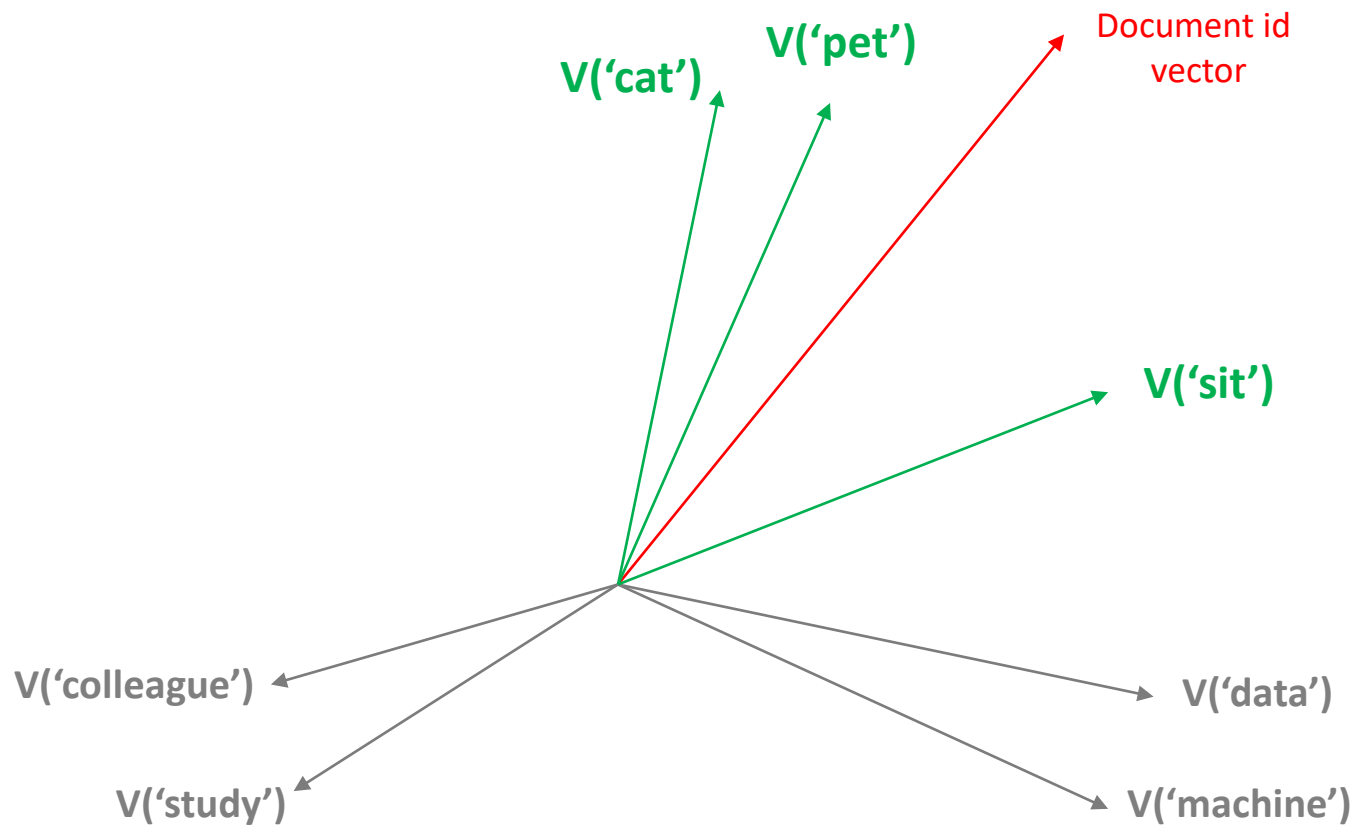


Document id를 단어처럼 취급하여
the, cat, sat 다음의 단어를 예측하는
Softmax regression을 학습

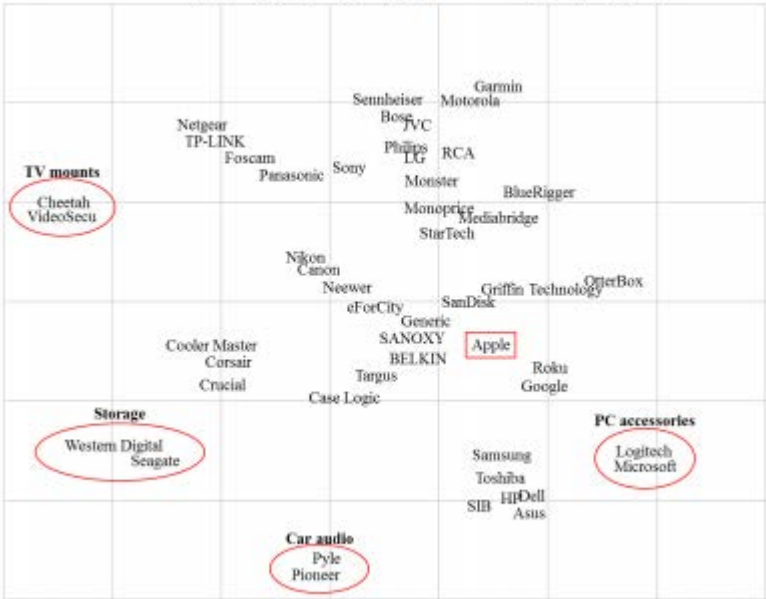
Doc2Vec

- Doc2vec은 한 문서에 등장한 단어 벡터들의 방향으로 document id vector를 위치시키도록 함

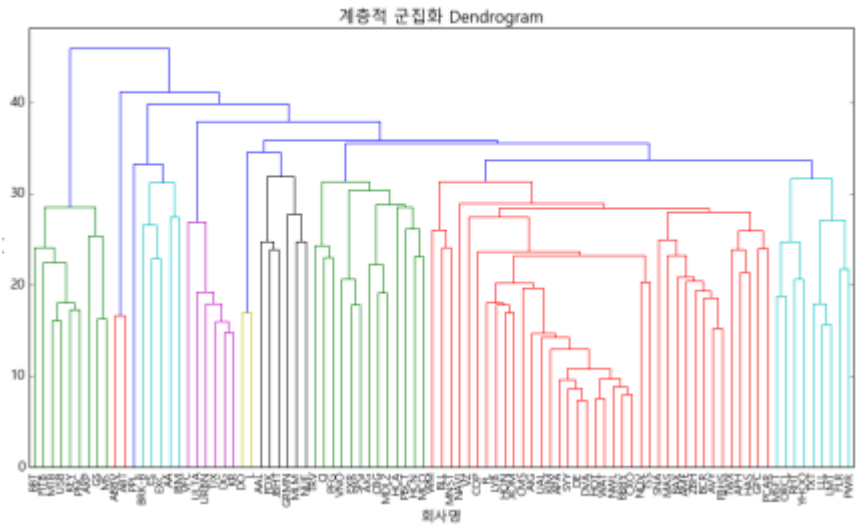
e.g) doc = my pet, a little cat sit on table



Amazon 제품 리뷰 문서를 이용한 Brand 관계 시각화



공시(10-K) 문서를 활용한 상장기업 clustering



Doc2vec 예시

- Word2vec 활용 정도
 - http://www.dbpia.co.kr/search/topSearch?startCount=0&collection=ALL&range=A&searchField=ALL&sort=RANK&query=2vec&srchOption=*&includeAr=false
- Doc2Vec과 Word2Vec을 활용한 Convolutional Neural Network 기반 한국어 신문 기사 분류

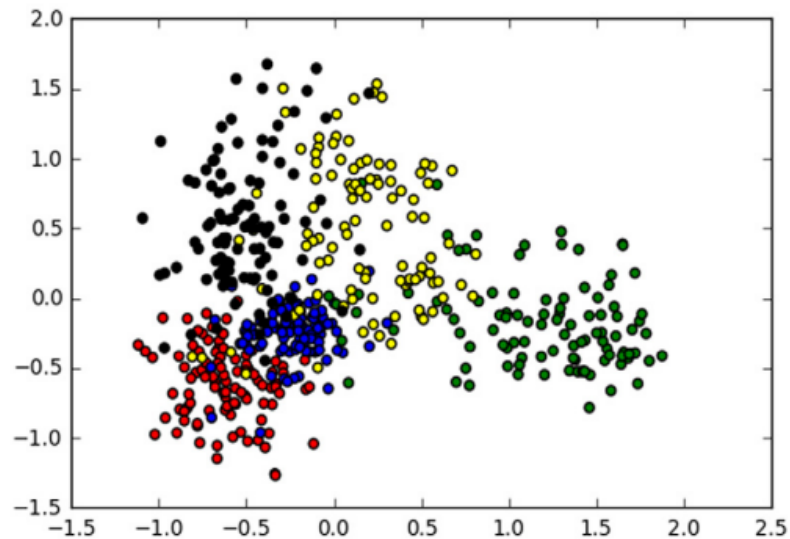


그림 2 범주별 문서 벡터의 산포도

Fig. 2 Scatter plot of document vectors per category

Doc2vec 예시

- 음식의 재료들을 고밀도 벡터 공간상 화학적 조합으로 임베딩하기 위한 방법론

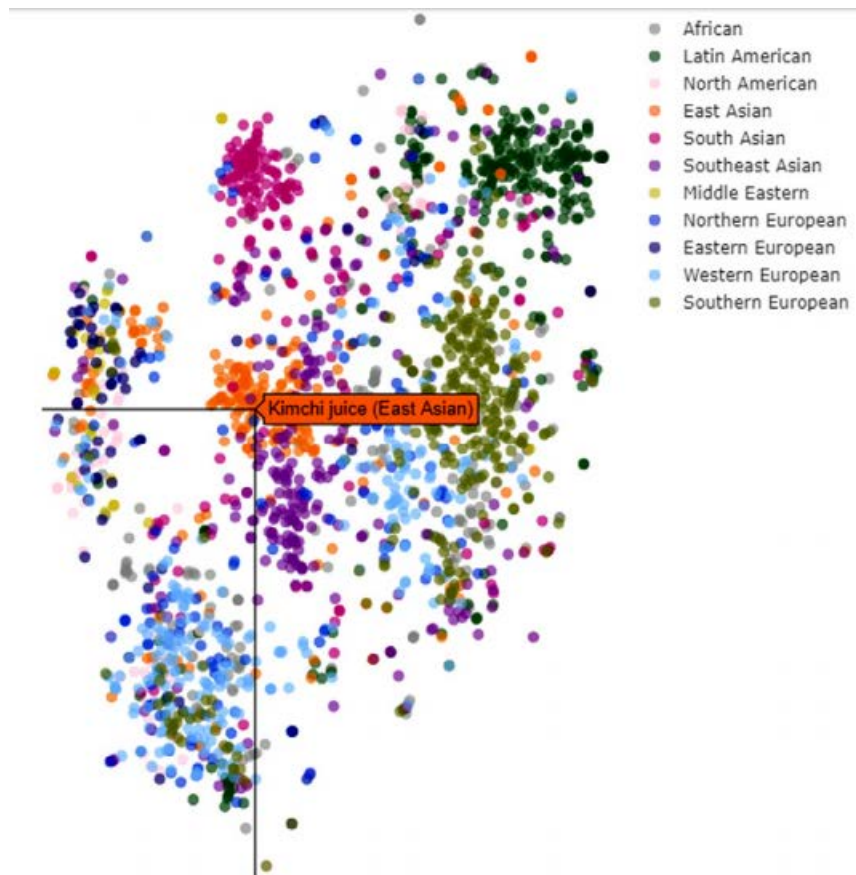
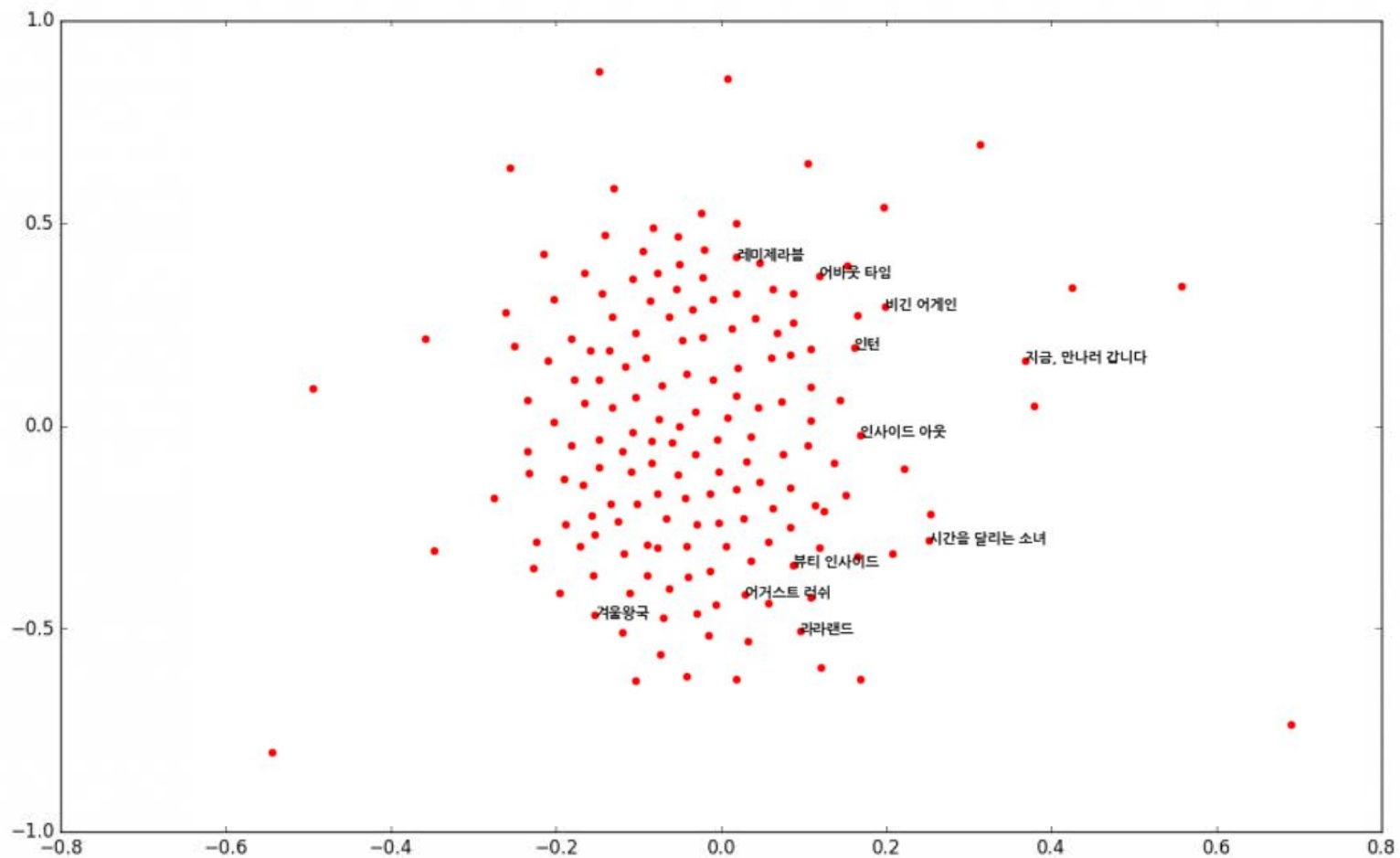


그림 1 Food2vec 시각화 결과

Fig. 1 Visualization results for Food2vec

Doc2vec 연습문제

- 아래 Doc2vec 결과에서 각 영화 (Document)의 임베딩 좌표를 구하고, 2개의 그룹으로 군집화해보세요.



Q&A

감사합니다.