

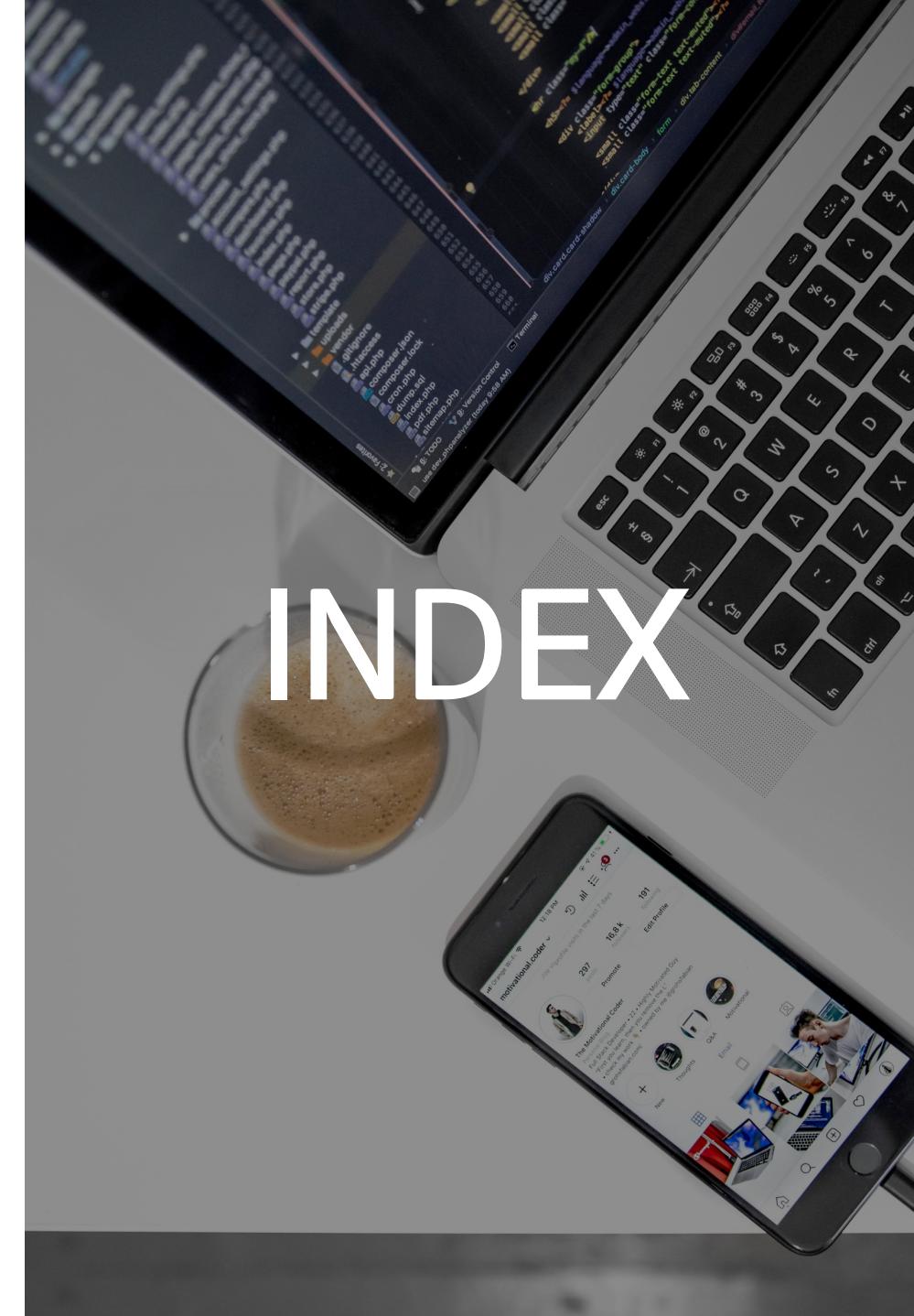


Summer NLP

Lec 01. Intro to NLP and PyTorch

파이토치로 배우는 자연어 처리 CHAPTER 1~2

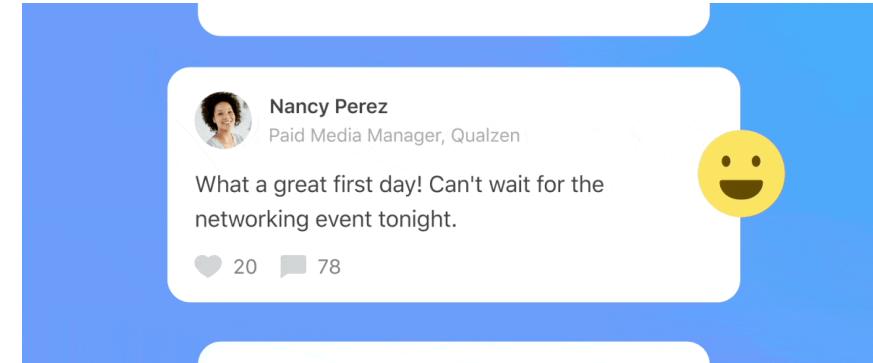
- | NLP에서의 머신러닝
- | 샘플과 타깃의 인코딩
- | PyTorch 기초
- | NLP 기술 빠르게 훑어보기



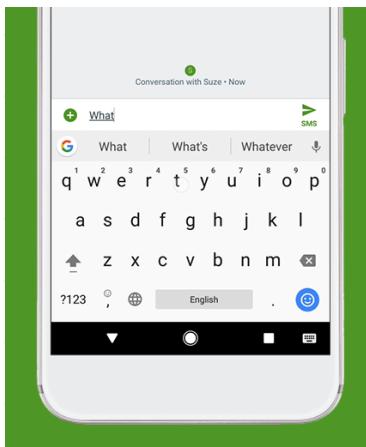
NLP에서의 머신러닝



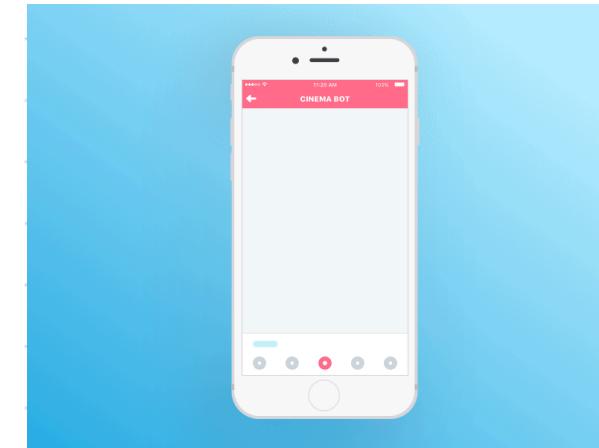
기계 번역 (Machine Translation)



감정 분석 (Sentiment Analysis)



단어 예측 (Word Prediction)



챗봇 (Chatbot)

| NLP에서의 머신러닝



NLP

언어학 지식과 상관없이 텍스트를 이해하는
통계적 방법을 사용해 실전 문제를 해결하는 기술

NLP에서의 머신러닝

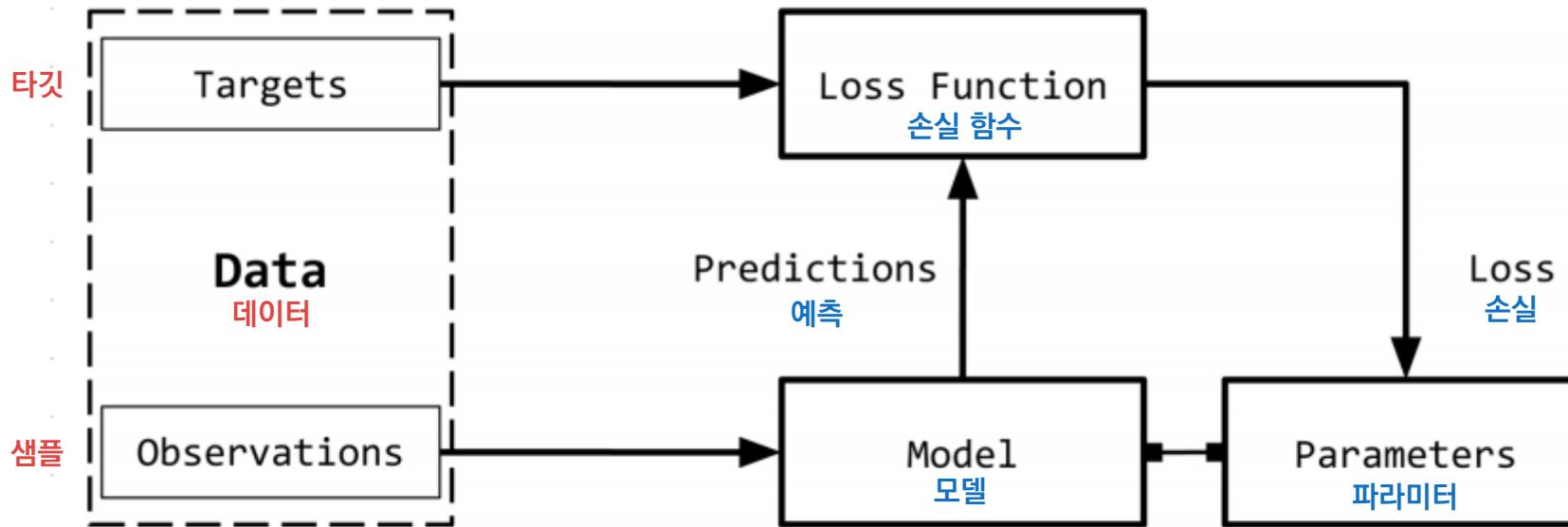


Figure 1-1. The supervised learning paradigm, a conceptual framework for learning from labeled input data.

Supervised Learning

본 책에서는 샘플과 타깃의 데이터로 학습하는 지도학습에 대해 다룹니다.

NLP에서의 머신러닝

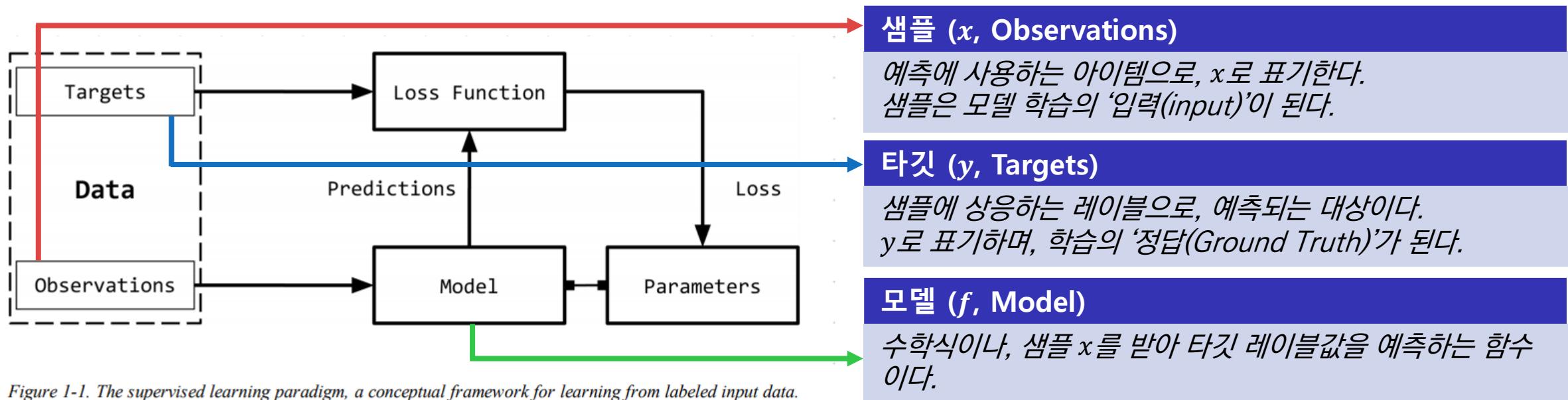
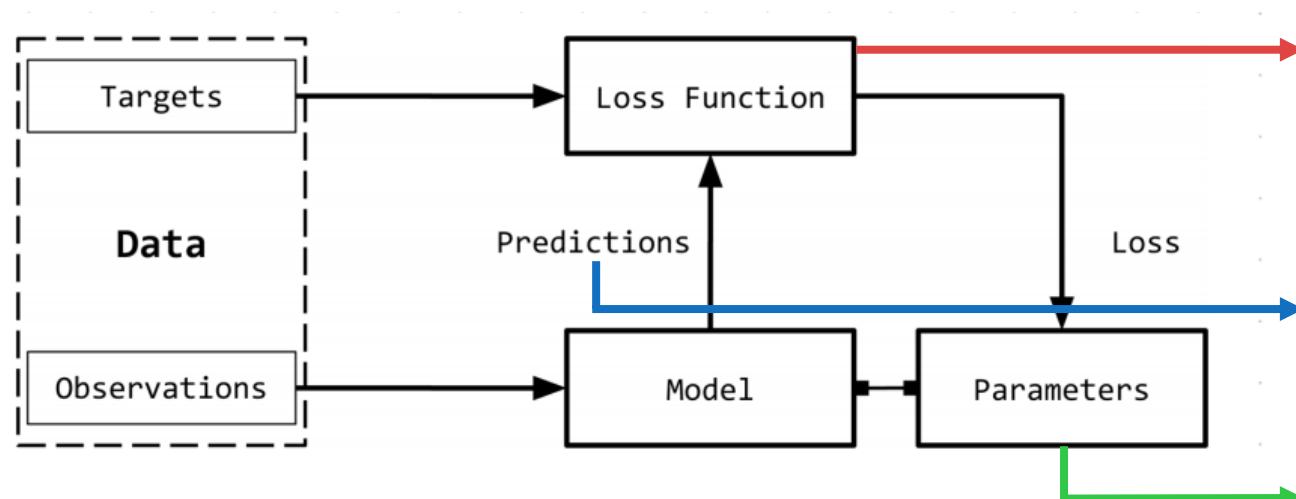


Figure 1-1. The supervised learning paradigm, a conceptual framework for learning from labeled input data.

Supervised Learning

본 책에서는 샘플과 타깃의 데이터로 학습하는 지도학습에 대해 다룹니다.

NLP에서의 머신러닝



손실함수 (L , Loss Function)

예측이 타깃과 얼마나 멀리 떨어져 있는지를 비교하는 함수이다. 타깃과 예측간의 스칼라값인 손실(loss)를 계산하는 함수이다. 낮은 손실일수록 예측을 더 잘하는 모델이다.

예측 (\hat{y} , Predictions)

모델이 추측하는 타깃값이다. \hat{y} 와 같이 *hat*으로 표현한다.

파라미터 (w , Parameters)

가중치(Weight)라고도 부르며, 모델을 이루는 값들이다.

Supervised Learning

본 책에서는 샘플과 타깃의 데이터로 학습하는 지도학습에 대해 다룹니다.

NLP에서의 머신러닝

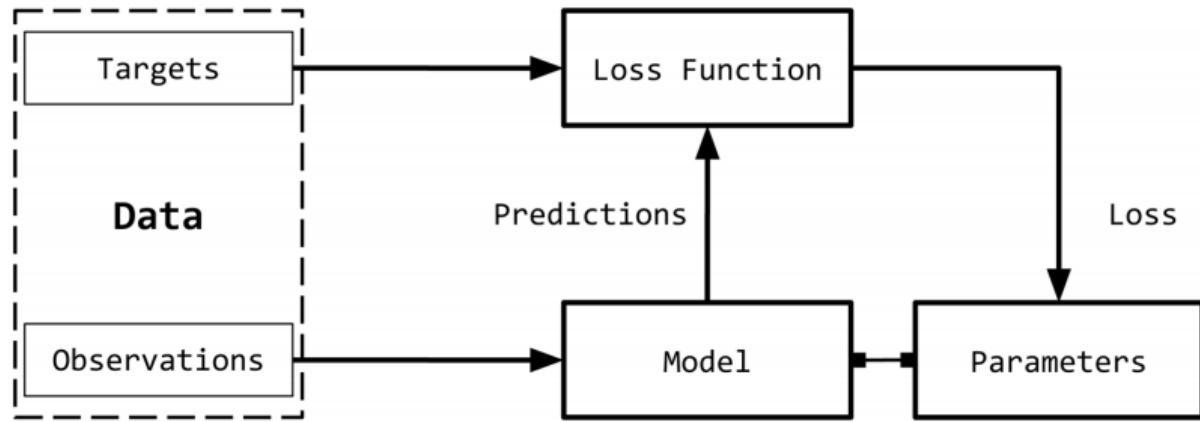


Figure 1-1. The supervised learning paradigm, a conceptual framework for learning from labeled input data.

$$D = \{X_i, y_i\}_{i=1}^N$$

Dataset : Observations, Targets

$$\hat{y} = f(X, w)$$

Prediction (Model with Parameter)

$$L(y, \hat{y}) = loss$$

Loss : Loss Function (Minimize Loss!)

Supervised Learning

본 책에서는 샘플과 타깃의 데이터로 학습하는 지도학습에 대해 다룹니다.

샘플과 타깃의 인코딩

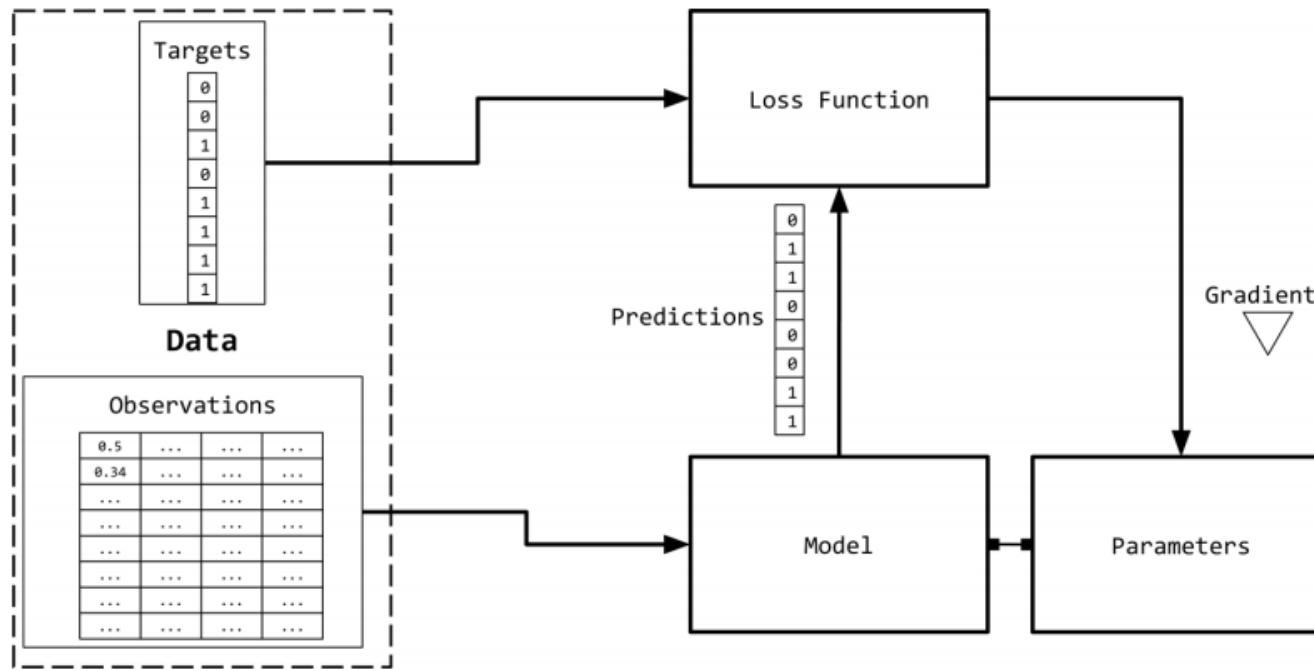


Figure 1-2. Observation and target encoding: The targets and observations from figure 1-1 are represented numerically as vectors, or tensors. This is collectively known as input “encoding.”

Encoding

모델이 자연어를 학습하기 위해서는 문장을 수치정보로 바꾸어야 합니다.

샘플과 타깃의 인코딩

Time flies like an arrow.

Fruit flies like a banana.



{time, fruit, flies, like, a, an, arrow, banana}

Vocabulary (어휘 사전)



	time	fruit	flies	like	a	an	arrow	banana
1_time	1	0	0	0	0	0	0	0
1_fruit	0	1	0	0	0	0	0	0
1_flies	0	0	1	0	0	0	0	0
1_like	0	0	0	1	0	0	0	0
1_a	0	0	0	0	1	0	0	0
1_an	0	0	0	0	0	1	0	0
1_arrow	0	0	0	0	0	0	1	0
1_banana	0	0	0	0	0	0	0	1

One-hot Representation

단어에 대한 Table을 만들고, 문장에 등장하는 상응하는 원소를 1로 설정

샘플과 타깃의 인코딩

	time	fruit	flies	like	a	an	arrow	banana
1_time	1	0	0	0	0	0	0	0
1_fruit	0	1	0	0	0	0	0	0
1_flies	0	0	1	0	0	0	0	0
1_like	0	0	0	1	0	0	0	0
1_a	0	0	0	0	1	0	0	0
1_an	0	0	0	0	0	1	0	0
1_arrow	0	0	0	0	0	0	1	0
1_banana	0	0	0	0	0	0	0	1

0
0
0
1
1
0
0
1

“Like a banana”



[0, 0, 0, 1, 1, 0, 0, 1]

Binary Encoding

문장에 존재하는 단어들은 1로 설정

샘플과 타깃의 인코딩

Example 1-1. Generating a “collapsed” one-hot or binary representation using scikit-learn

```
from sklearn.feature_extraction.text import CountVectorizer
import seaborn as sns

corpus = ['Time flies flies like an arrow.',
          'Fruit flies like a banana.']
one_hot_vectorizer = CountVectorizer(binary=True)
one_hot = one_hot_vectorizer.fit_transform(corpus).toarray()
sns.heatmap(one_hot, annot=True,
            cbar=False, xticklabels=vocab,
            yticklabels=['Sentence 2'])
```

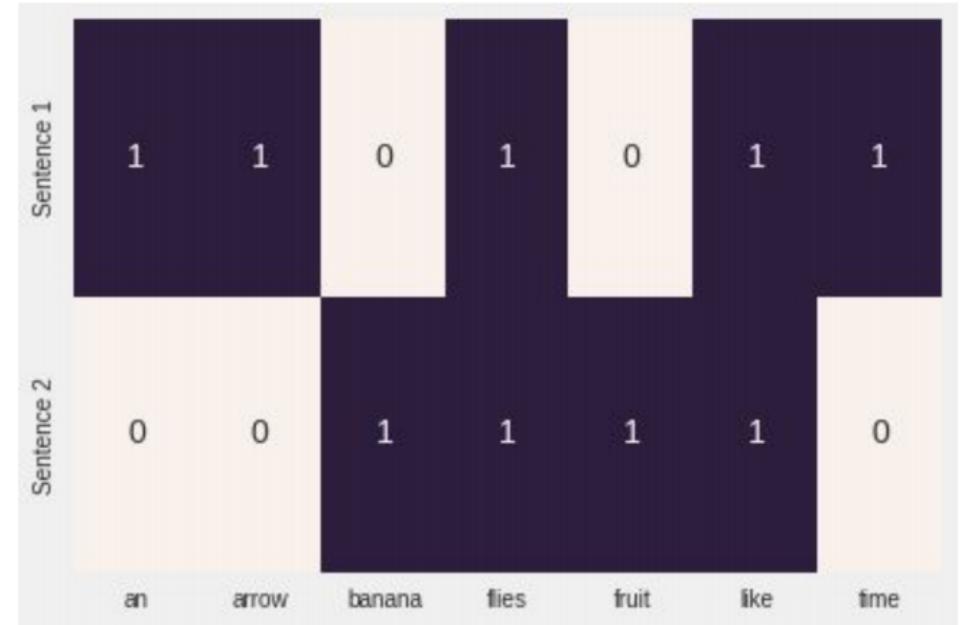


Figure 1-4. The collapsed one-hot representation generated by Example 1-1.

Binary Encoding

문장에 존재하는 단어들은 1로 설정

샘플과 타깃의 인코딩

	1	2	2	1	1			
1 _{time}	time	fruit	flies	like	a	an	arrow	banana
1 _{fruit}	1	0	0	0	0	0	0	0
1 _{flies}	0	1	0	0	0	0	0	0
1 _{like}	0	0	1	0	0	0	0	0
1 _a	0	0	0	1	0	0	0	0
1 _{an}	0	0	0	0	1	0	0	0
1 _{arrow}	0	0	0	0	0	0	1	0
1 _{banana}	0	0	0	0	0	0	0	1

1
2
2
1
1
0
0

“Fruit flies like time flies a fruit”



[1, 2, 2, 1, 1, 0, 0, 0]

TF Representation

Binary와는 달리, 문장에서 해당 단어의 개수를 센다.

| 샘플과 타깃의 인코딩

$$\text{IDF}(w) = \log \frac{N}{n_w}$$

N : 전체 문서의 개수
n_w : 단어 w를 포함한 문서의 개수

$$\text{TD - IDF} = \text{TF}(w) * \text{IDF}(w)$$

TF-IDF Representation

“희귀한 단어들은 문서의 특징을 잘 나타낸다”

샘플과 타깃의 인코딩

Example 1-2. Generating a TF-IDF representation using scikit-learn

```
from sklearn.feature_extraction.text import TfidfVectorizer
import seaborn as sns

tfidf_vectorizer = TfidfVectorizer()
tfidf = tfidf_vectorizer.fit_transform(corpus).toarray()
sns.heatmap(tfidf, annot=True, cbar=False, xticklabels=vocab,
            yticklabels= ['Sentence 1', 'Sentence 2'])
```

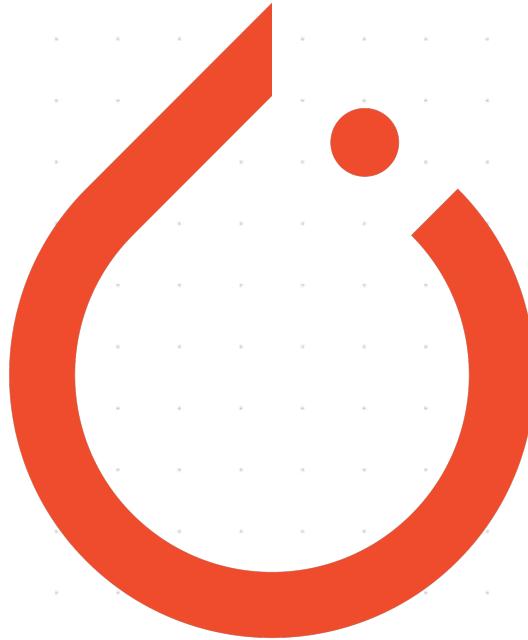


Figure 1-5. The TF-IDF representation generated by Example 1-2.

TF-IDF Representation

“희귀한 단어들은 문서의 특징을 잘 나타낸다”

| PyTorch 기초



PyTorch

계산그래프를 설계하고, 텐서 조작을 할 수 있도록 하는 라이브러리

PyTorch 기초

$$3 * 2 + 1$$

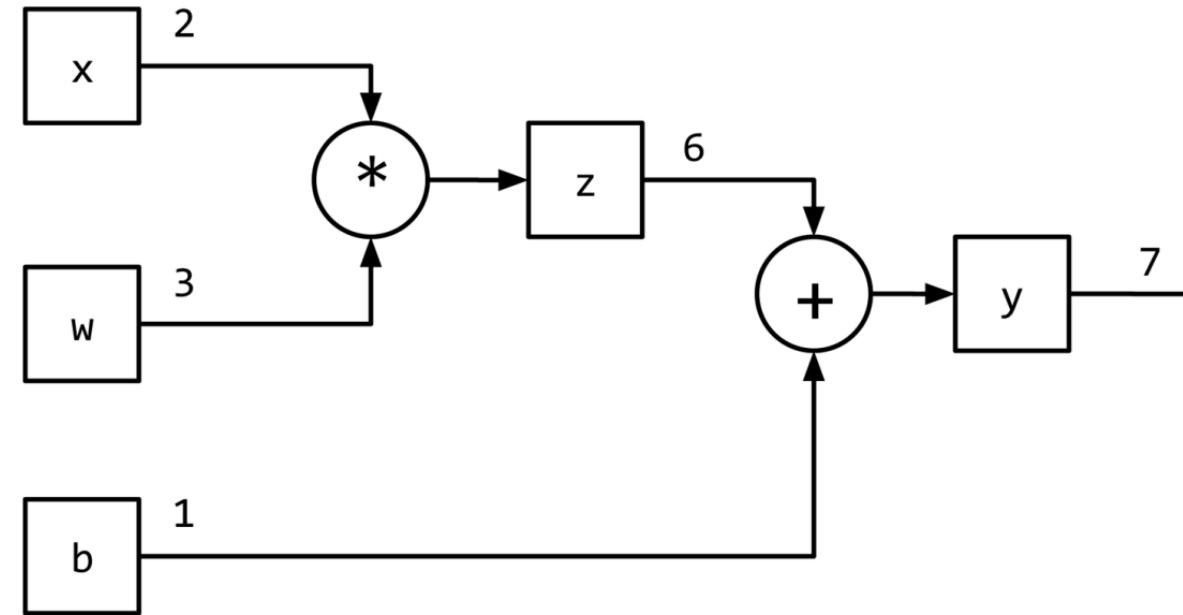


Figure 1-6. Representing $y = wx + b$ using a computational graph.

계산 그래프 (Computational Graph)

수학식을 추상적으로 모델링하여 자동 미분 등을 구현

PyTorch 기초

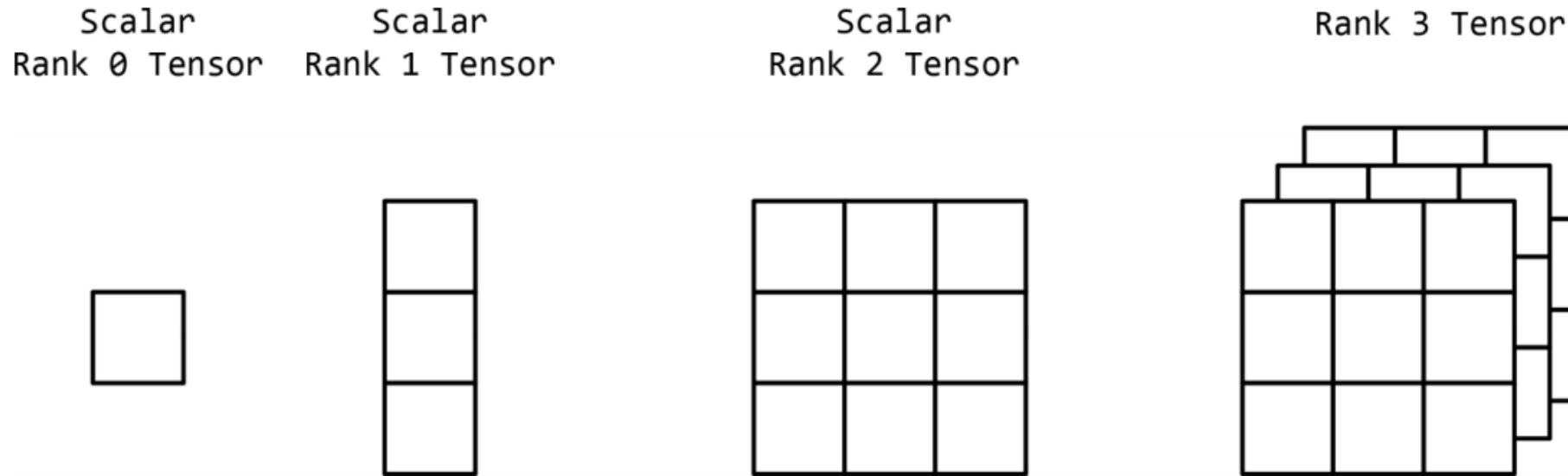
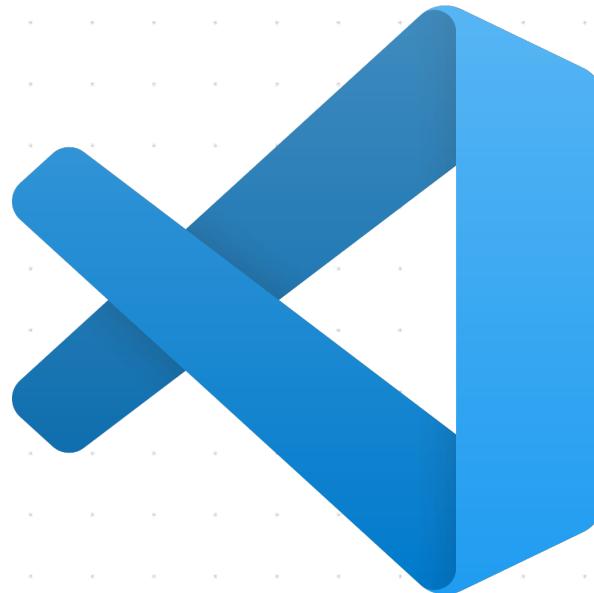


Figure 1-7. Tensors as a generalization of multidimensional arrays.

텐서 (Tensor)

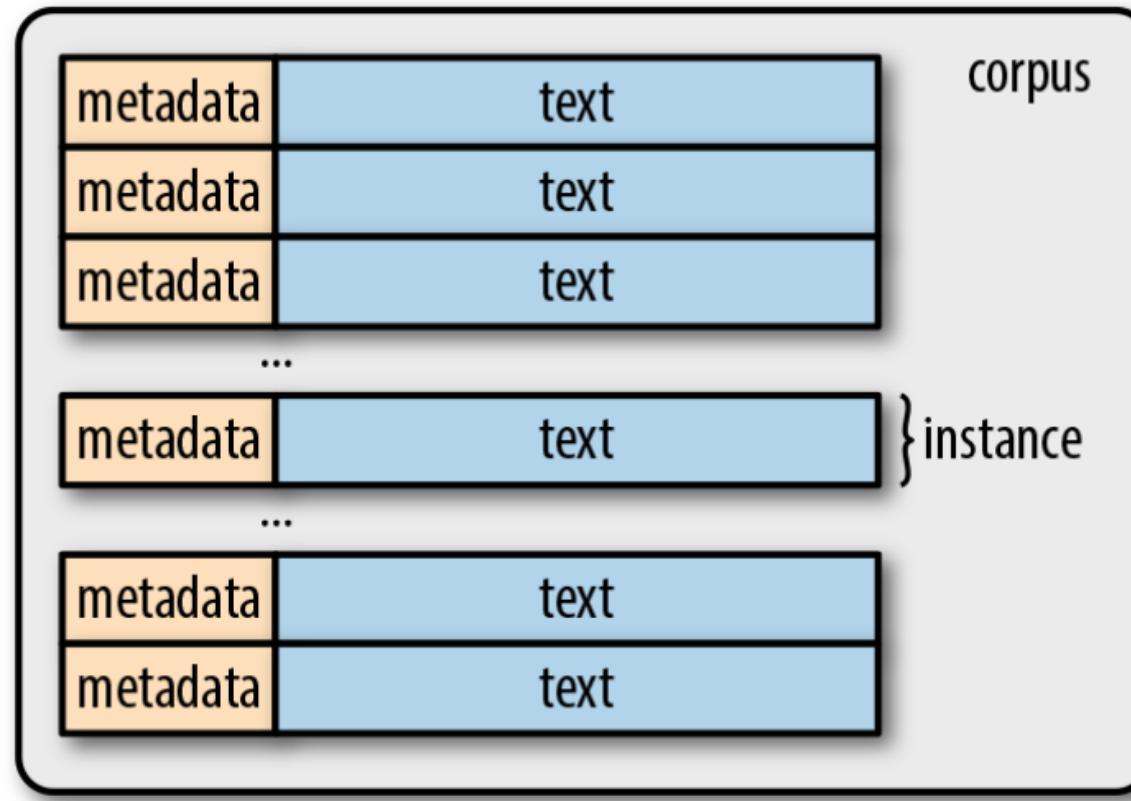
다차원의 데이터를 담은 수학적 객체. 0차는 Scalar, 1차는 Vector, 2차는 Matrix…

| PyTorch 기초



Lets Code!
With Colab

NLP 기술 빠르게 훑어보기



말뭉치 (Corpus)

NLP분야에서 사용되는 데이터셋으로, 원시 텍스트와 메타데이터로 이루어짐.

NLP 기술 빠르게 훑어보기

문장

“아버지가 가방에 들어가신다.”

토크나이징
(tokenizing)

토큰
(token)

아버지 가 가방 에 들어가 신다

토큰화 (Tokenization)

단어의 연속된 단위로 나누는 과정으로, 의미의 최소 단위로 나누게 된다.

NLP 기술 빠르게 훑어보기

Input[0]

```
import spacy
nlp = spacy.load('en')
text = "Mary, don't slap the green witch"
print([str(token) for token in nlp(text.lower())])
```

Output[0]

```
['mary', ',', 'do', "n't", 'slap', 'the', 'green', 'witch', '.']
```

Input[1]

```
from nltk.tokenize import TweetTokenizer
tweet=u"Snow White and the Seven Degrees
      #MakeAMovieCold@midnight:-)"
tokenizer = TweetTokenizer()
print(tokenizer.tokenize(tweet.lower()))
```

Output[1]

```
['snow', 'white', 'and', 'the', 'seven', 'degrees',
 '#makeamoviecold', '@midnight', ':-)']
```

토큰화 (Tokenization)

단어의 연속된 단위로 나누는 과정으로, 의미의 최소 단위로 나누게 된다.

NLP 기술 빠르게 훑어보기

```
from soynlp.tokenizer import MaxScoreTokenizer

scores = {w:s.cohesion_forward for w, s in words.items()}
tokenizer = MaxScoreTokenizer(scores=scores)
tokenizer.tokenize('맛있는짜파게티파스타일식초밥소바김볶다먹고싶어라일단김밥천국으로고고')

['맛있', '는', '짜파게티', '파스타', '일식', '초밥', '소바', '김볶', '다', '먹고', '싶어', '라', '일단', '김밥천국', '으로', '고고']
```

토큰화 (Tokenization)

단어의 연속된 단위로 나누는 과정으로, 의미의 최소 단위(형태소)로 나누게 된다.

NLP 기술 빠르게 훑어보기

Sample text with Stop Words	Without Stop Words
GeeksforGeeks – A Computer Science Portal for Geeks	GeeksforGeeks , Computer Science, Portal ,Geeks
Can listening be exhausting?	Listening, Exhausting
I like reading, so I read	Like, Reading, read

불용어 (Stopword)

관사, 전치사, 조사, 접미사와 같이 큰 의미가 없는 단어들. 내용어의 반대말

NLP 기술 빠르게 훑어보기

Input[0]

```
def n_grams(text, n):
    """
    takes tokens or text, returns a list of n-grams
    """
    return [text[i:i+n] for i in range(len(text)-n+1)]

cleaned = ['mary', ',', "n't", 'slap', 'green', 'witch', '.']
print(n_grams(cleaned, 3))
```

Output[0]

```
[['mary', ',', "n't"],
[',", "n't", 'slap'],
["n't", 'slap', 'green'],
['slap', 'green', 'witch'],
['green', 'witch', '.']]
```

N-gram

고정 길이(N)의 연속된 텍스트 토큰의 시퀀스

NLP 기술 빠르게 훑어보기

Example 2-3. Lemmatization: reducing words to their root forms

Input[0]

```
import spacy
nlp = spacy.load('en')
doc = nlp(u"he was running late")
for token in doc:
    print('{} --> {}'.format(token, token.lemma ))
```

Output[0]

```
he --> he
was --> be
running --> run
late --> late
```

표제어(Lemma)와 어간(Stem)

표제어는 단어의 원형, 어간은 수동으로 만든 규칙으로 단어를 공통 형태로 축소

NLP 기술 빠르게 훑어보기

Example 2-4. Part-of-speech tagging

Input[0]

```
import spacy
nlp = spacy.load('en')
doc = nlp(u"Mary slapped the green witch.")
for token in doc:
    print('{} - {}'.format(token, token.pos_))
```

Output[0]

```
Mary - PROPN
slapped - VERB
the - DET
green - ADJ
witch - NOUN
. - PUNCT
```

품사(POS) 태깅

단어 각각을 품사로 분류하여 태그를 붙임

NLP 기술 빠르게 훑어보기

Example 2-5. Noun Phrase (NP) chunking

Input[0]

```
import spacy
nlp = spacy.load('en')
doc = nlp(u"Mary slapped the green witch.")
for chunk in doc.noun_chunks:
    print ('{} - {}'.format(chunk, chunk.label_))
```

Output[0]

```
Mary - NP
the green witch - NP
```

첨크 나누기

구문 분석으로, 명사 동사 형용사와 같은 문법 요소로 구성된 고차원의 단위를 유도

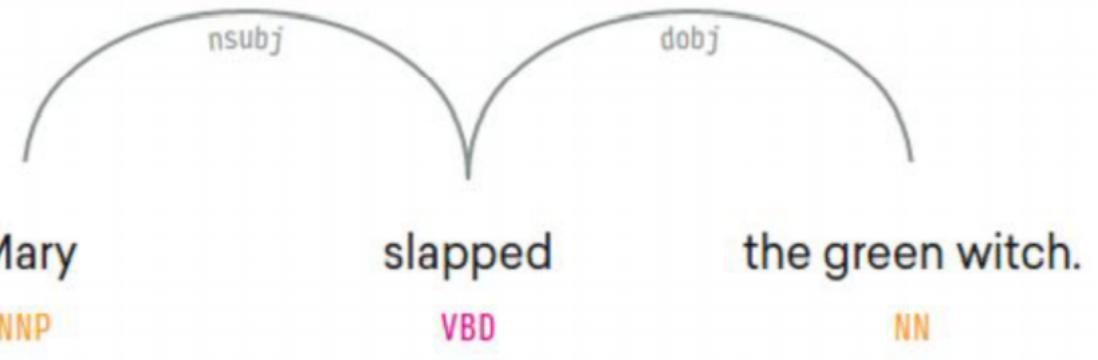
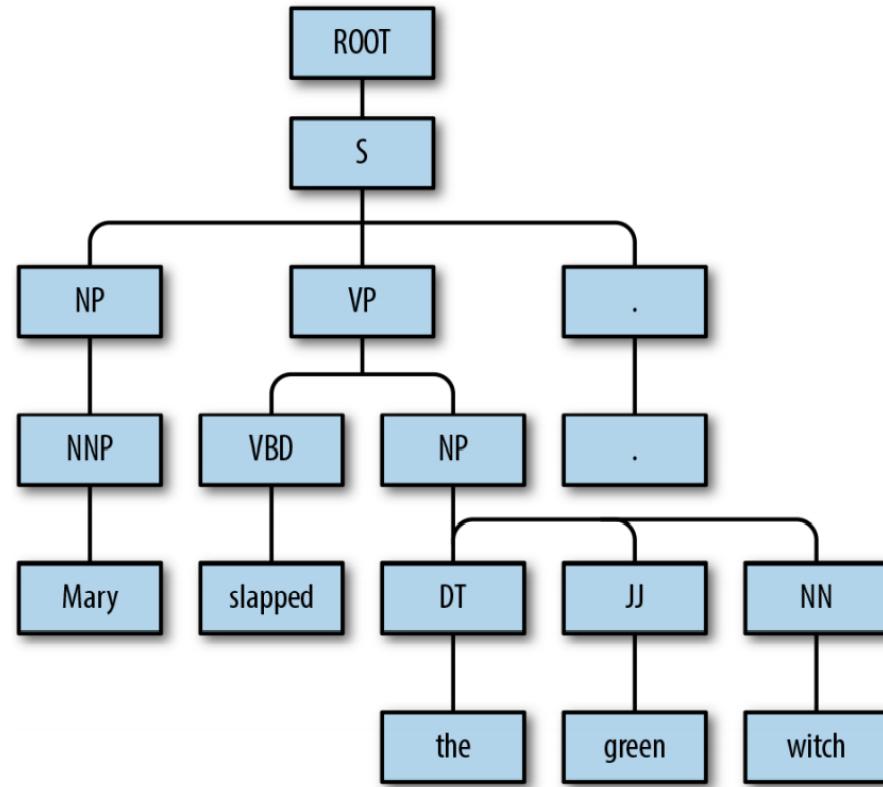
NLP 기술 빠르게 훑어보기

John PERSON was born in Chicken GPE , Alaska GPE , and studies at Cranberry Lemon University ORG .

개체명 인식

사람, 장소, 회사, 약 이름과 같은 실제 세상의 개념을 의미하는 문자열

NLP 기술 빠르게 훑어보기



문장 구조 분석

왼쪽은 구성 구문 분석, 오른쪽은 의존 구문 분석



Thank You