Preprocessing

(Part of Speech & TF-IDF)

Ko, Youngjoong

Dept. of Computer Science & Engineering,
SKK University

Index

- **Introduction**
- **How to build colab environment**
- **Part of Speech**
- **TF-IDF**
- **Assignment**

Introduction

❖ Introduction

- The assignment is to implement the functions calculating normalized TF-IDF for the given texts.

- In this PDF, we will explain the ‘Part-Of-Speech’ and ‘normalized TF-IDF’, and will give you guidelines for the assignment.

- Before explaining the assignment, we will show how to build colab environment.

Build Environment

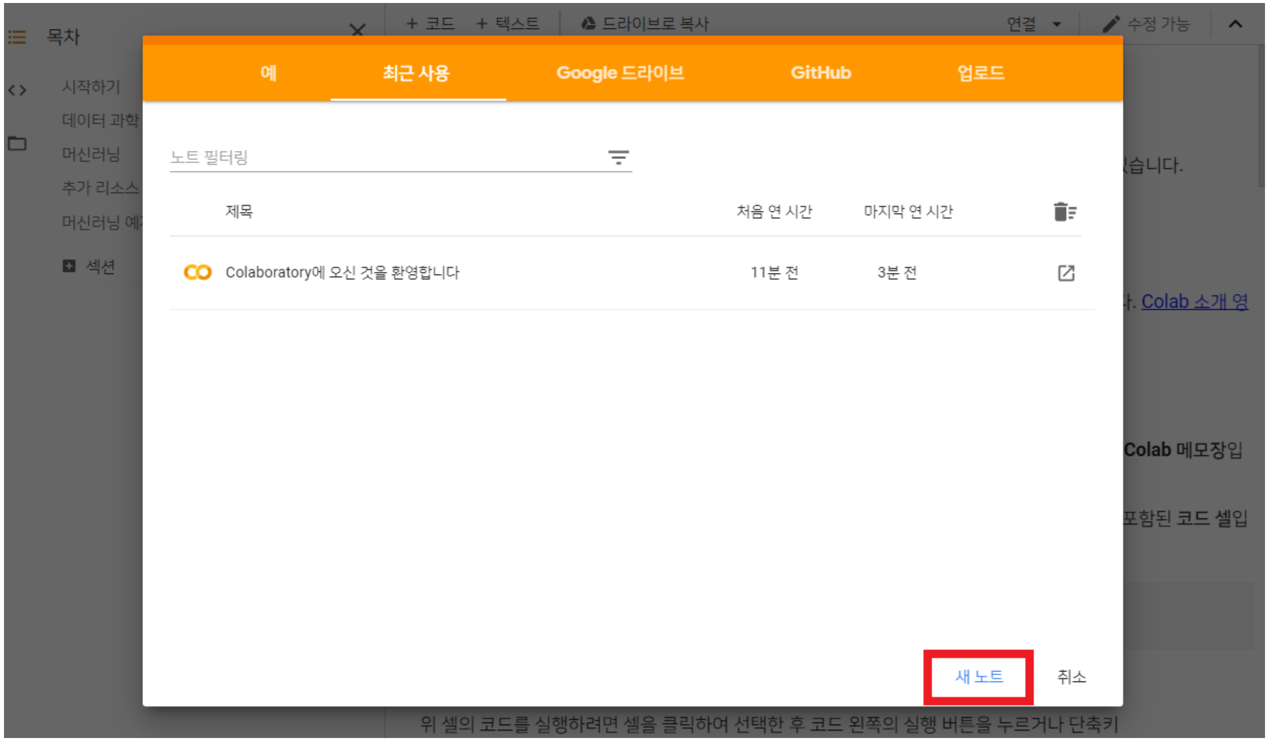
- **Google, ‘Colab’**
  - Google Cloud Development Environment.
  - This allows you to access a free GPU for up to 12 hours at a time.
  - Need a personal Google account.

colab

(*https://colab.research.google.com/*)

Build Environment

- **Google, 'Colab'**
  - Create New Notebook

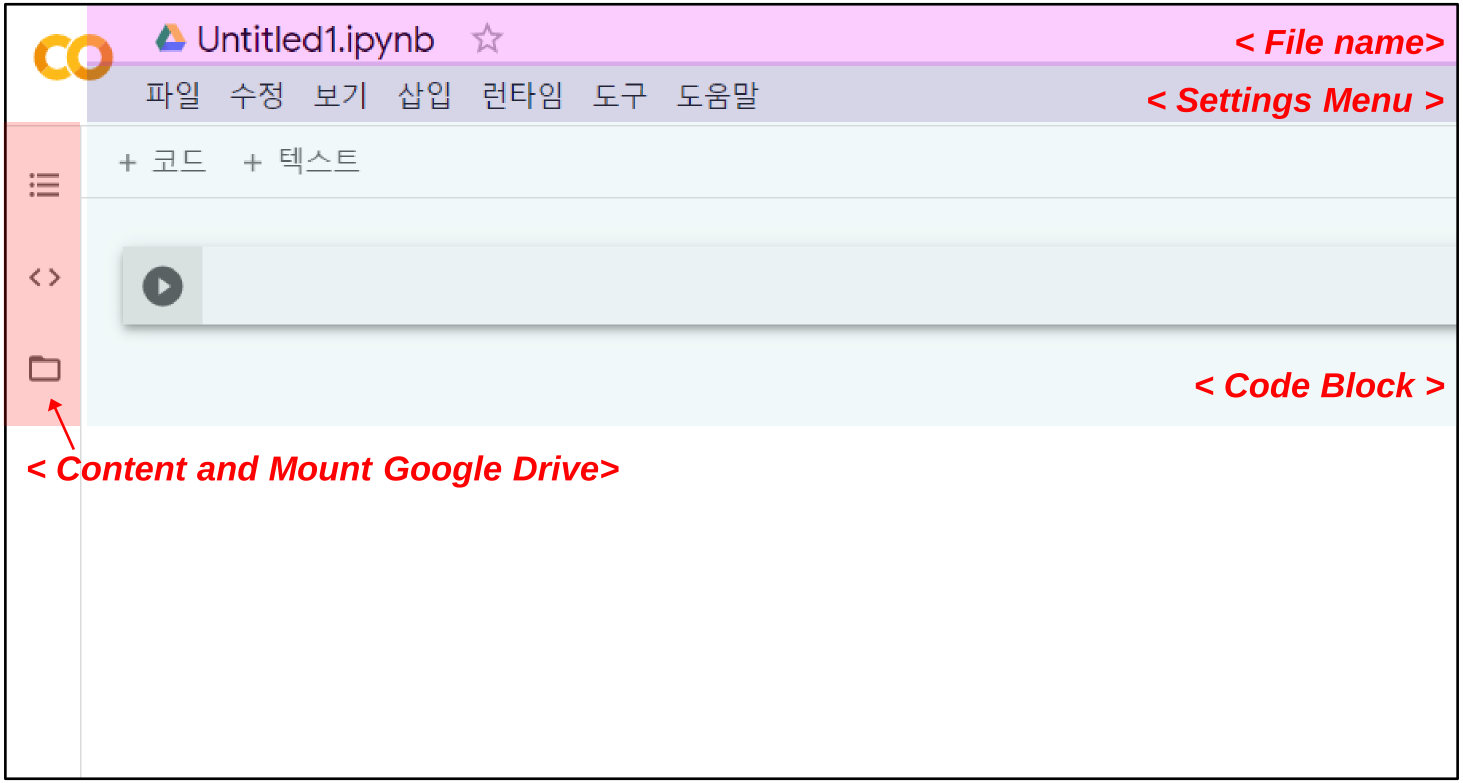


제목	처음 연 시간	마지막 연 시간
Colaboratory에 오신 것을 환영합니다	11분 전	3분 전

위 셀의 코드를 실행하려면 셀을 클릭하여 선택한 후 코드 왼쪽의 실행 버튼을 누르거나 단축키

Build Environment

- **Google, 'Colab'**
  - New Notebook



< File name>

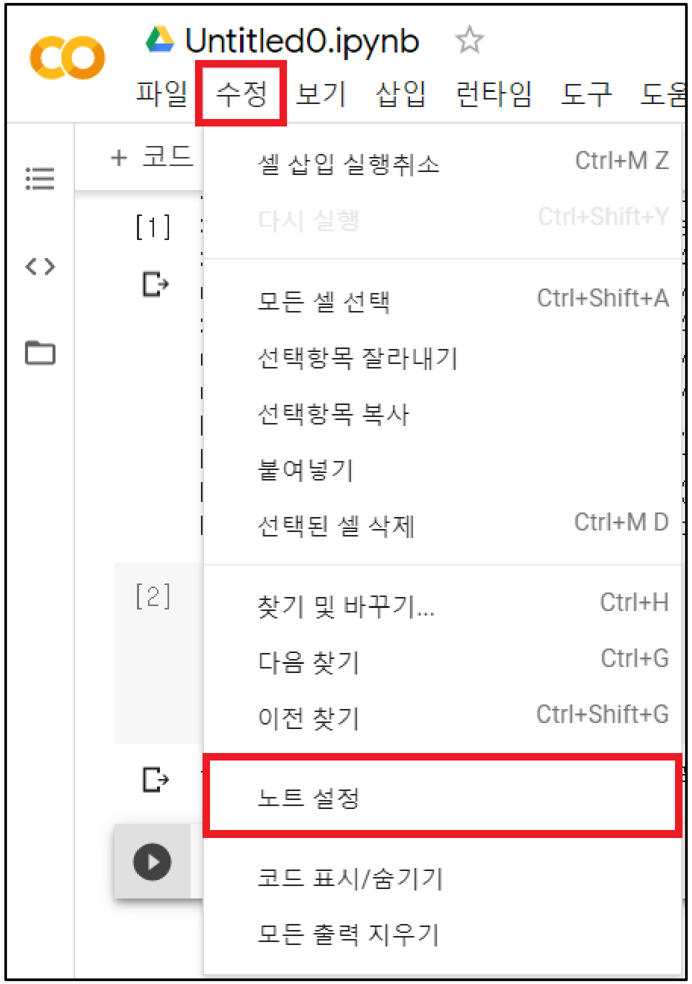
< Settings Menu >

< Code Block >

< Content and Mount Google Drive>

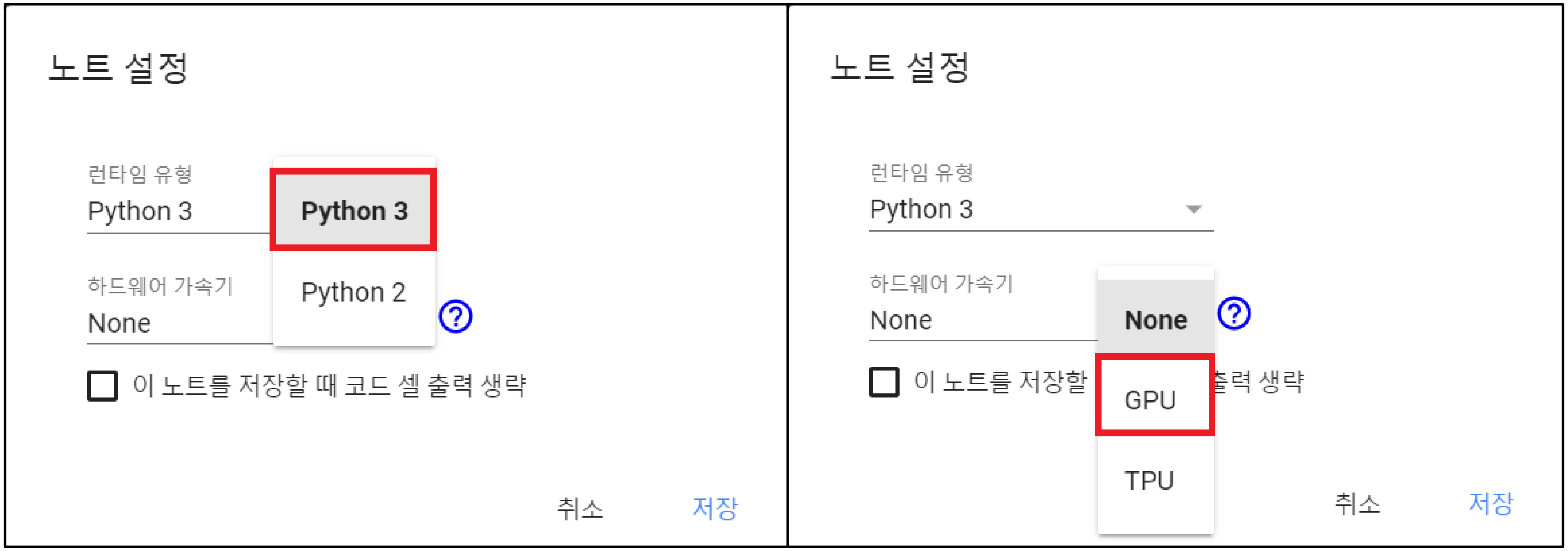
Build Environment

- **Google, 'Colab'**
  - Notebook Settings



Build Environment

- **Google, 'Colab'**
  - Notebook Settings
    - Python 3
    - GPU



노트 설정

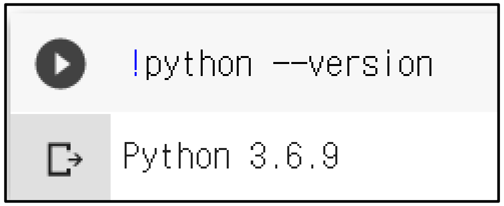
런타임 유형
Python 3

하드웨어 가속기
None

□ 이 노트를 저장할 때 코드 셀 출력 생략

Build Environment

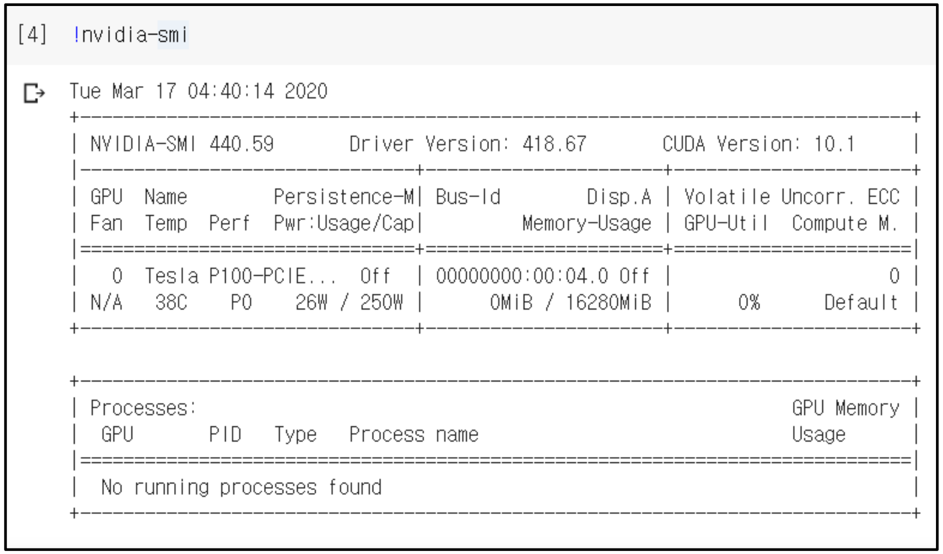
- **Google, 'Colab'**
  - Check Notebook settings
    - Python



```
!python --version
```

```
Python 3.6.9
```

    - GPU



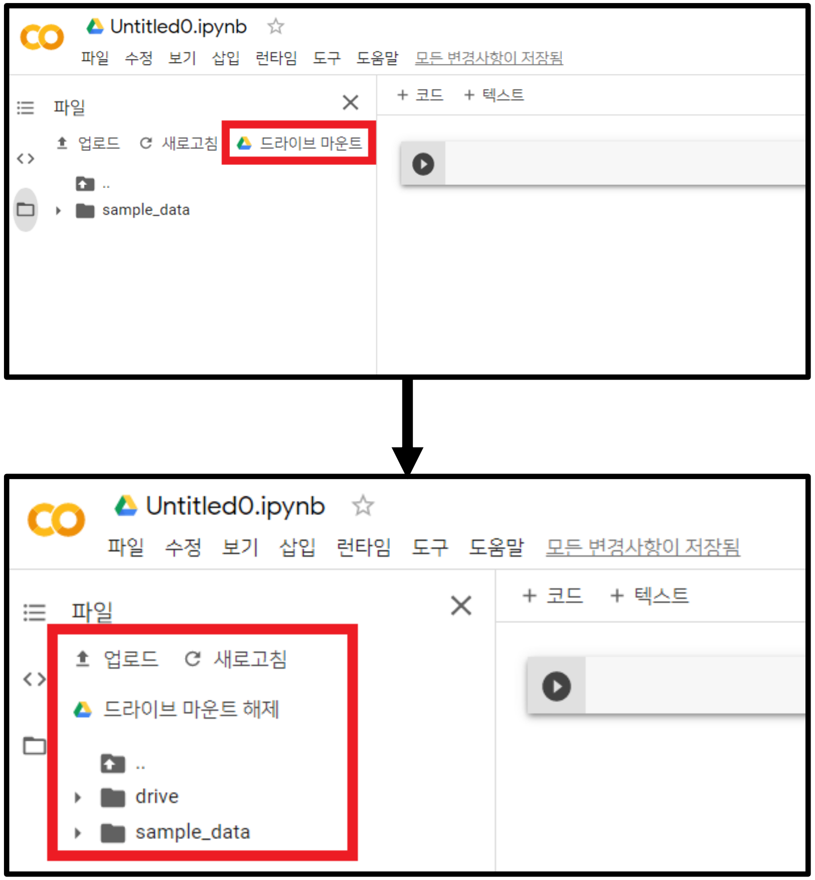
```
[4]  !nvidia-smi
```

```
Tue Mar 17 04:40:14 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.59       Driver Version: 418.67       CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   38C    P0    26W / 250W |      0MiB / 16280MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

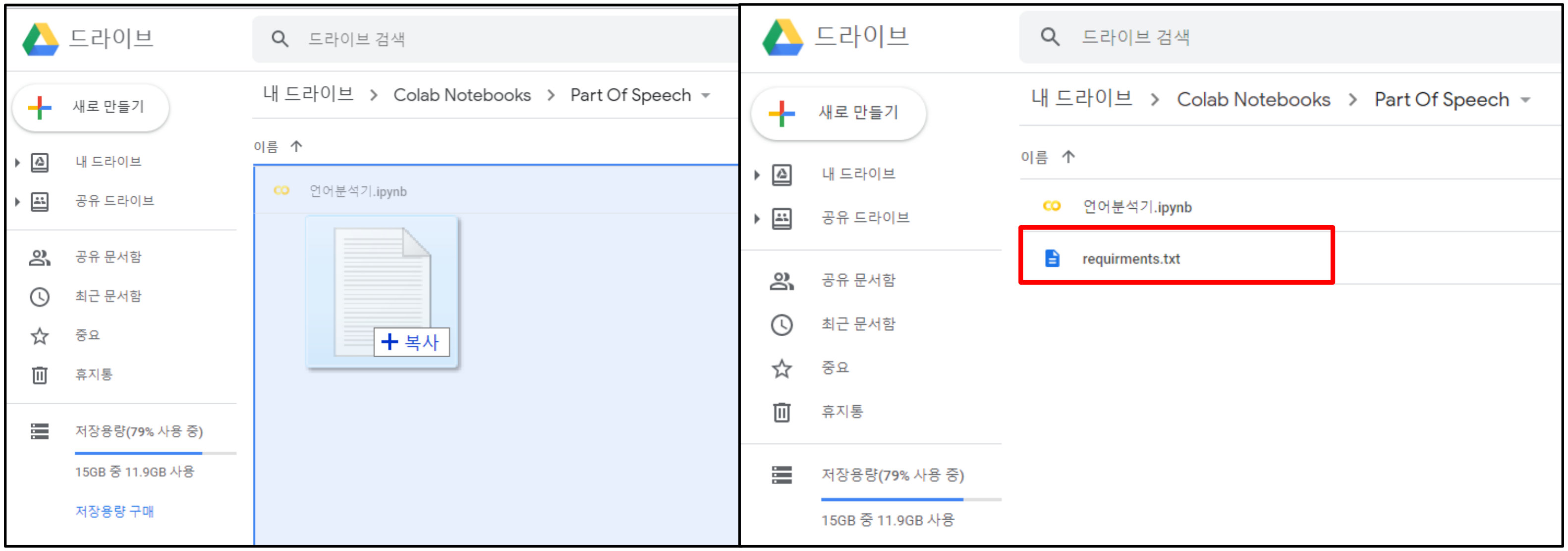
Build Environment

- **Google, ‘Colab’**
  - Google Drive Mount


CO
Untitled0.ipynb ☆
파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨
파일
업로드 새로고침 드라이브 마운트
..
sample_data
+ 코드 + 텍스트
CO
Untitled0.ipynb ☆
파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨
파일
업로드 새로고침
드라이브 마운트 해제
..
drive
sample_data
+ 코드 + 텍스트

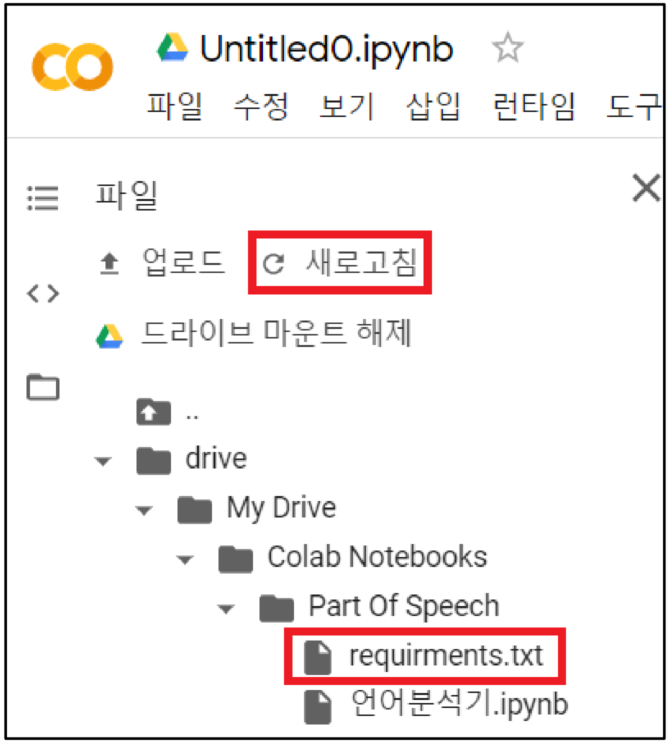
Build Environment

- **Google, ‘Colab’**
  - Upload data to google drive



Build Environment

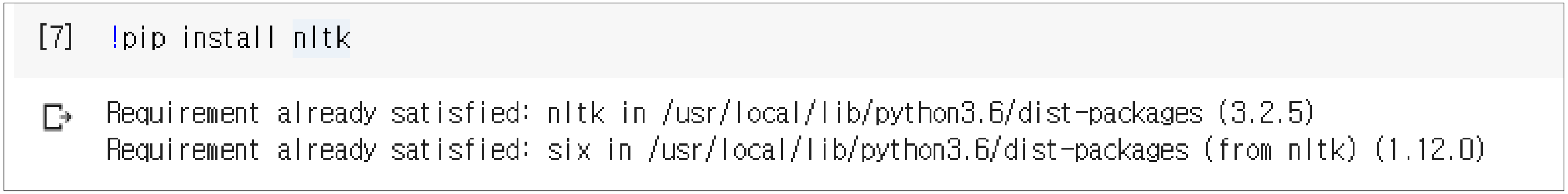
- **Google, 'Colab'**
  - After refresh, check the drive update history



Build Environment

❖ How to install NLTK in Python

- ➢ Install NLTK



```
[7]  !pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.6/dist-packages (3.2.5)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from nltk) (1.12.0)
```

Given Dataset

- Data Format : .json

- Data Example :

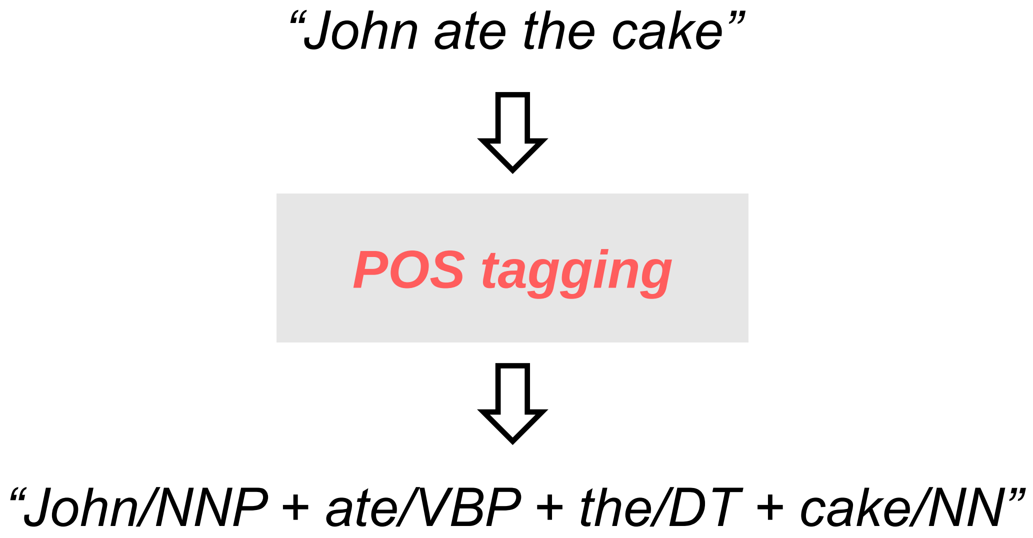
id	paragraph	label
1	Perhaps President Trump is right …	finance
2	From Town & Country Pippa Middleton …	entertainment
…		
1000	New York Mets prospect Tim Tebow …	sports

- The number of Labels : 5 (sports, entertainment, lifestyle, finance and tv)

Part of Speech

- **What is POS (Part-Of-Speech) tag?**
  - A POS tag is a label assigned to each word in a text to indicate the part-of-speech.
    Ex) Noun (NN, NNS, NNP), Verb (VB, VBD, VBG)

- **What is POS (Part-Of-Speech) tagging?**



“John ate the cake”

POS tagging

“John/NNP + ate/VBP + the/DT + cake/NN”

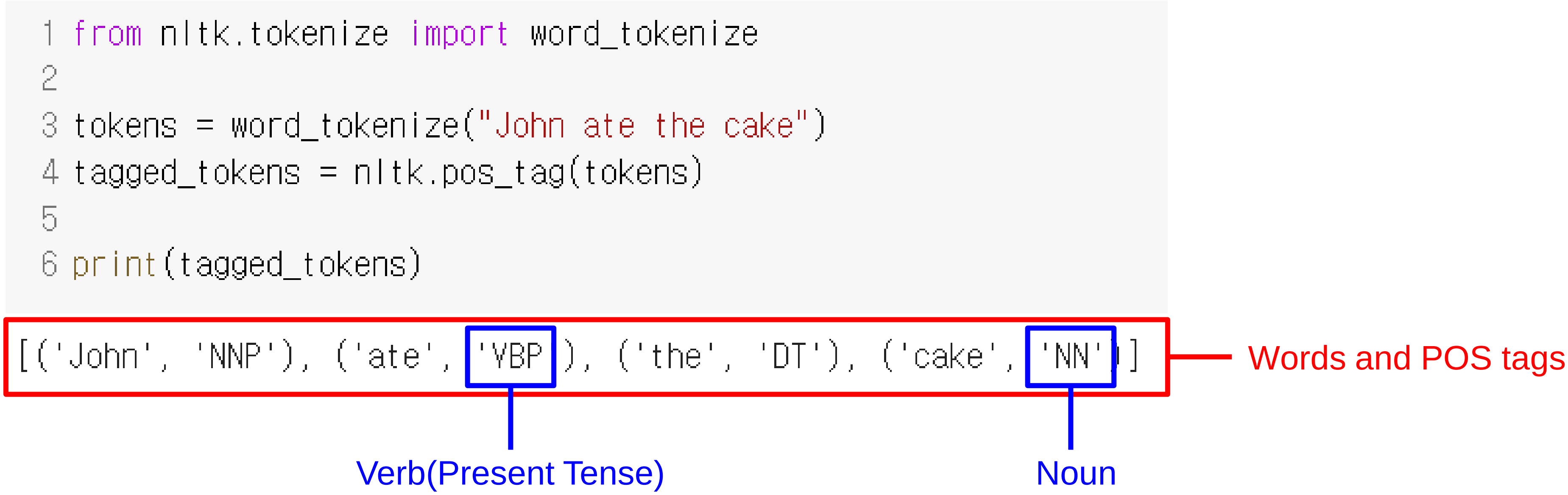
Part of Speech

POS tags of NLTK library for POS tagging

Tag	Description	Example
CC	coordinating conjunction	and
CD	cardinal number	1, third
DT	determiner	the
EX	existential there	*there* is
FW	foreign word	d’hoevre
IN	preposition/subordinating conjunction	in, of, like
JJ	adjective	big
JJR	adjective, comparative	bigger
JJS	adjective, superlative	biggest
LS	list marker	1)
MD	Modal	could, will
NN	noun, singular or mass	Door
NNS	noun plural	Doors
NNP	proper noun, singular	John
NNPS	proper noun, plural	Vikings
PDT	Predeterminer	*both* the boys
POS	possessive ending	friend*‘s*
PRP	personal pronoun	I, he, it
PRP$	possessive pronoun	my, his
RB	Adverb	however, usually
RBR	adverb, comparative	Better
RBS	adverb, superlative	Best
RP	Particle	give *up*
TO	To	*to* go, *to* him
UH	Interjection	Uhhuhhuhh
VB	verb, base form	Take
VBD	verb, past tense	Took
VBG	verb, gerund/present participle	Taking
VBN	verb, past participle	Taken
VBP	verb, sing. present, non-3d	Take
VBZ	verb, 3rd person sing. Present	Takes
WDT	wh-determiner	Which
WP	wh-pronoun	who, what
WP$	possessive wh-pronoun	Whose
WRB	wh-abverb	where, when

Part of Speech

❖ Example of NLTK library



```python
from nltk.tokenize import word_tokenize

tokens = word_tokenize("John ate the cake")
tagged_tokens = nltk.pos_tag(tokens)

print(tagged_tokens)
```

```
[('John', 'NNP'), ('ate', 'VBP'), ('the', 'DT'), ('cake', 'NN')]
```

TF-IDF

What is TF (Term Frequency)?

- TF shows how frequent a word occurs in a document.
- **Example of TF calculation**
  - Doc0 : i/PRP am/VBP a/DT boy/NN
  - Doc1 : i/PRP am/VBP a/DT girl/NN
  - Doc2 : who/WP is/VBZ a/DT boy/NN

	a/DT	am/VBP	boy/NN	girl/NN	i/PRP	is/VBZ	who/WP
Doc_0	1	1	1	0	1	0	0
Doc_1	1	1	0	1	1	0	0
Doc_2	1	0	1	0	0	1	1

The limitation of TF

- Frequently used words like 'and’, 'the', 'a', ‘i', and 'you' will be weighted highly due to their frequent usage, even though they are not important.

TF-IDF

What is IDF (Inverse Document Frequency)?

- DF is used to determine whether a term is common or rare across all documents.
- Common words have less information compared to the ones that occur rarely.
- IDF is a way of damping the weights of common terms and increasing the weights of those that occur infrequently.
- **Example of IDF calculation**
  - $log_2 \frac{N}{df_t}$: Inverse value of DF
  - N : Total number of document
  - df_t : The number of documents that contain the term t (df_t)

  ex) am/VBP = $log_2 \frac{3}{2} = 0.58$, girl/NN= $log_2 \frac{3}{1} = 1.58$

	a/DT	am/VBP	boy/NN	girl/NN	i/PRP	is/VBZ	who/WP
IDF	0	0.58	0.58	1.58	0.58	1.58	1.58

TF-IDF

- **What is TF-IDF (Term Frequency-Inverse Document Frequency)?**

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t,D)$$

Term frequency		Document frequency		Normalization	
l (logarithm)	$1 + \log(tf_{t,d})$	n (no)	1	n (none)	1
n (natural)	$tf_{t,d}$	t (idf)	$\log \frac{N}{df_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2 + w_2^2 + \cdots + w_M^2}}$
a (augmented)	$0.5 + \frac{0.5 \times tf_{t,d}}{\max_t tf_{t,d}}$	p (prob idf)	$\max\left\{0, \log \frac{N - df_t}{df_t}\right\}$	u (pivoted unique)	$\frac{1}{u}$
b (boolean)	$\begin{cases} 1 & if \;\; tf_{t,d} > 0 \\ 0 & otherwise \end{cases}$			b (byte size)	$\frac{1}{CharLength^{\alpha}}, \alpha < 1$
L (log ave)	$\frac{1 + \log(tf_{t,d})}{1 + \log(ave_{t \in d}(tf_{t,d}))}$				

We will use these fomulas

TF-IDF

Normalized TF-IDF Example

- TF

	a/DT	am/VBP	boy/NN	girl/NN	I/PRP	is/VBZ	who/WP
Doc_0	1	1	1	0	1	0	0
Doc_1	1	1	0	1	1	0	0
Doc_2	1	0	1	0	0	1	1

- IDF

	a/DT	am/VBP	boy/NN	girl/NN	i/PRP	is/VBZ	who/WP
IDF	0	0.58	0.58	1.58	0.58	1.58	1.58

- TF-IDF

	a/DT	am/VBP	boy/NN	girl/NN	i/PRP	is/VBZ	who/WP
Doc_0	0	0.58	0.58	0	0.58	0	0
Doc_1	0	0.58	0	1.58	0.58	0	0
Doc_2	0	0	0.58	0	0	1.58	1.58

TF-IDF

❖ Normalized TF-IDF Example

➢ Normalization of TF-IDF

- $Normalized\ Doc_0 = \frac{1}{\sqrt{(0+0.58^2+0.58^2+0+0.58^2+0+0)}} \times Doc_0$

=

Doc_0	0	$\frac{0.58}{\sqrt{1.00}}$	$\frac{0.58}{\sqrt{1.00}}$	0	$\frac{0.58}{\sqrt{1.00}}$	0	0

- $Normalized\ Doc_1 = \frac{1}{\sqrt{(0+0.58^2+\ 0+1.58^2+\ 0.58^2+0+0)}} \times Doc_1$

=

Doc_1	0	$\frac{0.58}{\sqrt{3.17}}$	0	$\frac{1.58}{\sqrt{3.17}}$	$\frac{0.58}{\sqrt{3.17}}$	0	0

- $Normalized\ Doc_2 = \frac{1}{\sqrt{(0+0+0.58^2+\ 0+0+1.58^2+1.58^2)}} \times Doc_2$

=

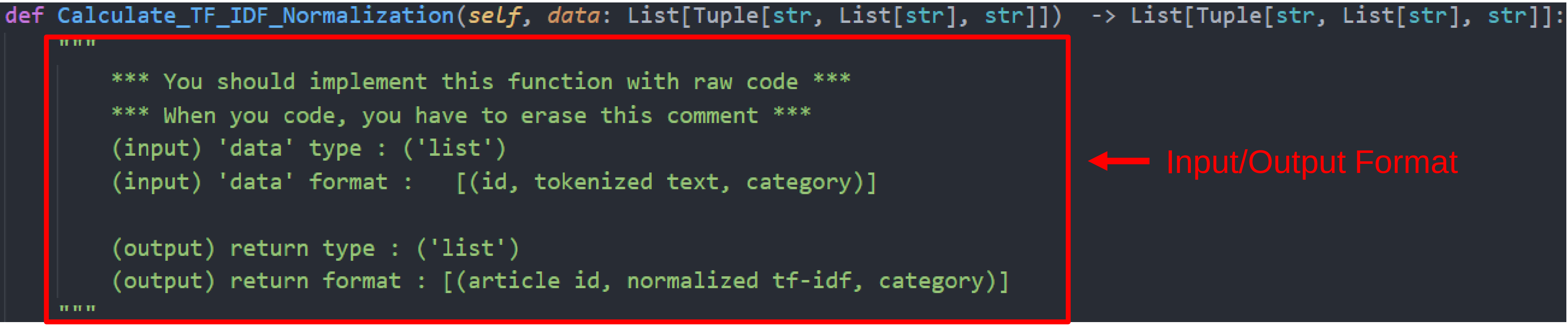
Doc_2	0	0	$\frac{0.58}{\sqrt{5.33}}$	0	0	$\frac{1.58}{\sqrt{5.33}}$	$\frac{1.58}{\sqrt{5.33}}$

Assignment

- **Assignment**
  - Implement the functions calculating normalized TF-IDF according to above explanation.
  - We provide ‘train.json’, ‘test.json’, ‘HW3_main.py’ and ‘HW3_util.py’.
  - You have to implement 1 function (Calculate_TF_IDF_Normalization) which is in ‘HW3_main.py’ (please see page 25 for specific formats).
  - Use ‘Python3’ and ‘Google Colab’.
  - Do not import any additional library except those that are already imported.
  - Plagiarized submissions (Copied codes) will be scored 0.

Assignment

- **‘HW3_main.py’**
  - We provide input/output format of three functions in 'HW3_main.py’ as the comments, so you can refer it



```
def Calculate_TF_IDF_Normalization(self, data: List[Tuple[str, List[str], str]])  -> List[Tuple[str, List[str], str]]:
    """
        *** You should implement this function with raw code ***
        *** When you code, you have to erase this comment ***
        (input) 'data' type : ('list')
        (input) 'data' format :    [(id, tokenized text, category)]

        (output) return type : ('list')
        (output) return format : [(article id, normalized tf-idf, category)]
    """
```

Input/Output Format

  - Sort final TF-IDF values in alphabetical order (a-z)
    - Ex)

	d	a	c	b
Doc0	0	0.41	0.41	0

	a	b	c	d
Doc0	0.41	0	0.41	0

  - Round the final TF-IDF values which are the output of the function ‘Calculate_TF_IDF_Normalization’ to the second digit after the decimal point using 'round()' function

ex) 0.4054… ⇒ 0.41 / 1.0986… ⇒ 1.1

Assignment

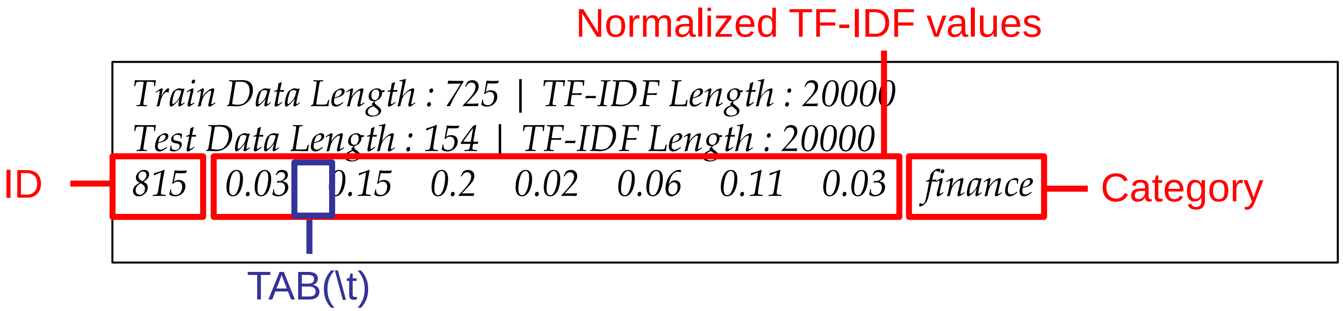
❖ Submission File

1) Python code file (.py) (python version 3.x)
   - Format: “hw3_StudentName_StudentID_main.py”.

     - Ex) “hw3_ 홍길동_2020000000_main.py”

2) TEXT file (.txt)
   - Format: “hw3_StudentName_StudentID.txt”.

❖ Format of the text file that you have to submit



```
Train Data Length : 725 | TF-IDF Length : 20000
Test Data Length : 154 | TF-IDF Length : 20000
815	0.03	0.15	0.2	0.02	0.06	0.11	0.03	finance
```

Thank you for your attention!

고 영 중 (Ko, Youngjoong)

http://nlp.skku.edu/