

The background of the slide features a white humanoid robot with black joints and a black base. It is holding a blue digital tablet in its right hand, which displays the binary code "0110000110". The robot's left hand is pointing towards the viewer. The robot is positioned on the right side of the slide, while the text is on the left.

Overview

Exception Handling

최성철 교수

Director of TEAMLAB

**프로그램 사용할 때
일어나는 혼한 일들**



<http://jino.me/1837>

[생각해보기]

프로그램 사용할 때 일어나는 오류들

- 주소를 입력하지 않고 배송 요청
- 저장도 안 했는데 컴퓨터 전원이 나감
- 게임 아이템 샀는데 게임에서 튕김

→ 예상치 못한 많은 일(**예외**)들이 생김

Exception

- 1) 예상 가능한 예외
- 2) 예상이 불가능한 예외

예상 가능한 예외

- 발생 여부를 사전에 인지할 수 있는 예외
- 사용자의 잘못된 입력, 파일 호출시 파일 없음
- 개발자가 반드시 명시적으로 정의 해야함

예상 불가능한 예외

- 인터프리터 과정에서 발생하는 예외, 개발자 실수
- 리스트의 범위를 넘어가는 값 호출, 정수 0으로 나눔
- 수행 불가시 인터프리터가 자동 호출

예외 처리 (Exception Handling)

- 예외가 발생할 경우 후속 조치 등 대처 필요

- 1) 없는 파일 호출 → 파일 없음을 알림
- 2) 게임 이상 종료 → 게임 정보 저장

프로그램 = 제품, 모든 잘못된상황에 대처가 필요

Exception Handling

TEAMLAB

Human knowledge belongs to the world.



Implementation

Exception Handling

최성철 교수
Director of TEAMLAB

예외 처리

Exception Handling

파이썬의 예외 처리

try ~ except 문법

try:

예외 발생 가능 코드

except <Exception Type>:

예외 발생시 대응하는 코드

파이썬의 예외 처리 예제

- 0으로 숫자를 나눌 때 예외처리 하기

```
for i in range(10):
    try:
        print(10 / i)
    except ZeroDivisionError:
        print("Not divided by 0")
```

Exception의 종류

- Built-in Exception: 기본적으로 제공하는 예외

Exception 이름	내용
IndexError	List의 Index 범위를 넘어갈 때
NameError	존재하지 않은 변수를 호출 할 때
ZeroDivisionError	0으로 숫자를 나눌 때
ValueError	변환할 수 없는 문자/숫자를 변환할 때
FileNotFoundException	존재하지 않는 파일을 호출할 때

파이썬의 예외 처리 예제

- 예외 정보 표시하기

```
for i in range(10):
    try:
        print(10 / i)
    except ZeroDivisionError as e:
        print(e)
        print("Not divided by 0")
```

else 구문

try ~ except ~ else

try:

예외 발생 가능 코드

except <Exception Type>:

예외 발생시 동작하는 코드

else:

예외가 발생하지 않을 때 동작하는 코드

else 구문 예시

try ~ except ~ else

```
for i in range(10):
    try:
        result = 10 / i
    except ZeroDivisionError:
        print("Not divided by 0")
    else:
        print(10 / i)
```

finally 구문

try ~ except ~ finally

try:

예외 발생 가능 코드

except <Exception Type>:

예외 발생시 동작하는 코드

finally:

예외 발생 여부와 상관없이 실행됨

finally 구문 예시

try ~ except ~ finally

```
try:  
    for i in range(1, 10):  
        result = 10 // i  
        print(result)  
except ZeroDivisionError:  
    print("Not divided by 0")  
finally:  
    print("종료되었습니다.")
```

raise 구문

필요에 따라 강제로 Exception을 발생

```
raise <Exception Type>(예외정보)
```

```
while True:  
    value = input("변환할 정수 값을 입력해주세요")  
    for digit in value:  
        if digit not in "0123456789":  
            raise ValueError("숫자값을 입력하지 않으셨습니다")  
    print("정수값으로 변환된 숫자 -", int(value))
```

Assert 구문

특정 조건에 만족하지 않을 경우 예외 발생

assert 예외조건

```
def get_binary_nmubmer(decimal_number):
    assert isinstance(decimal_number, int)
    return bin(decimal_number)

print(get_binary_nmubmer(10))
```



Human knowledge belongs to the world.

Overview

File

최성철 교수
Director of TEAMLAB

01100
00110

**컴퓨터를 실행할 때
가장 기본이 되는 단위**

컴퓨터를 실행할 때
가장 기본이 되는 단위

파일

[생각해보기]

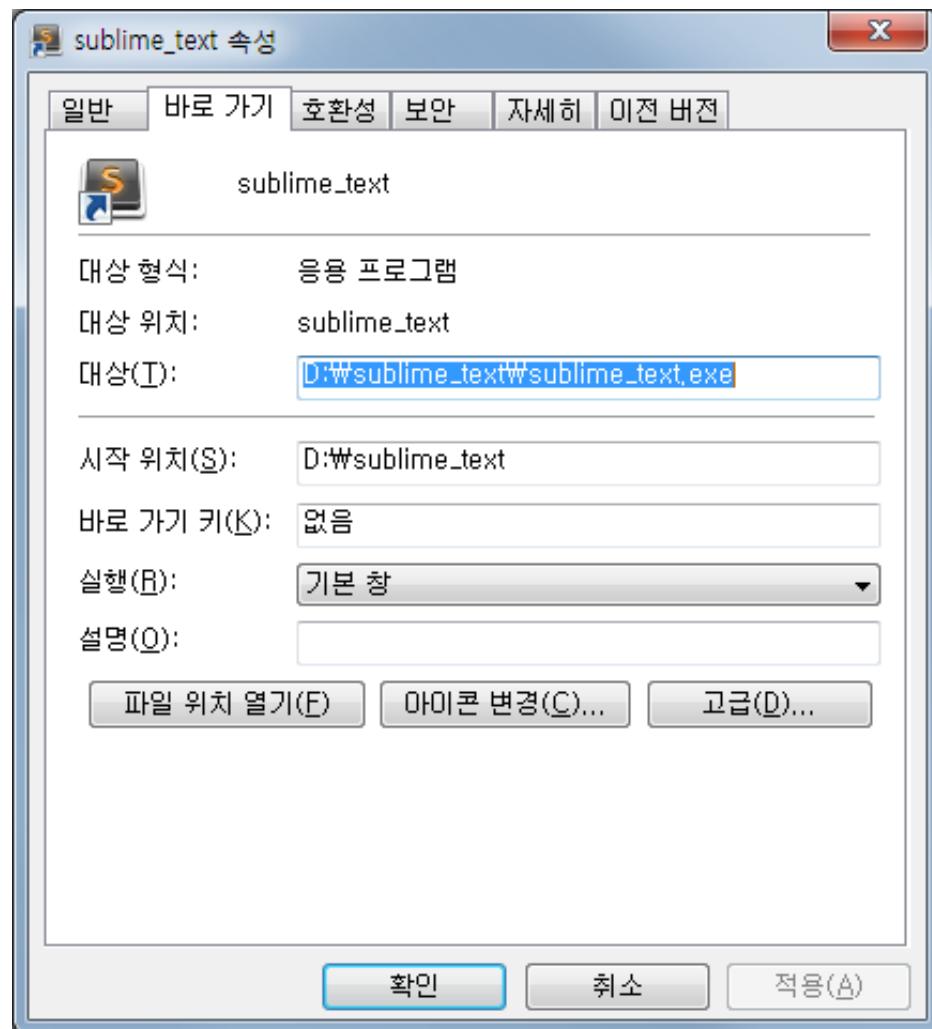
우리는 어떻게 프로그램을 시작하나?



보통은 이렇게 생긴 아이콘을 누른다!

그러나 실제로는
아이콘이 아닌 “실행 파일”을 실행시키는 것
아이콘을 클릭하고 오른쪽 마우스 클릭 “속성”을 선택해 볼 것

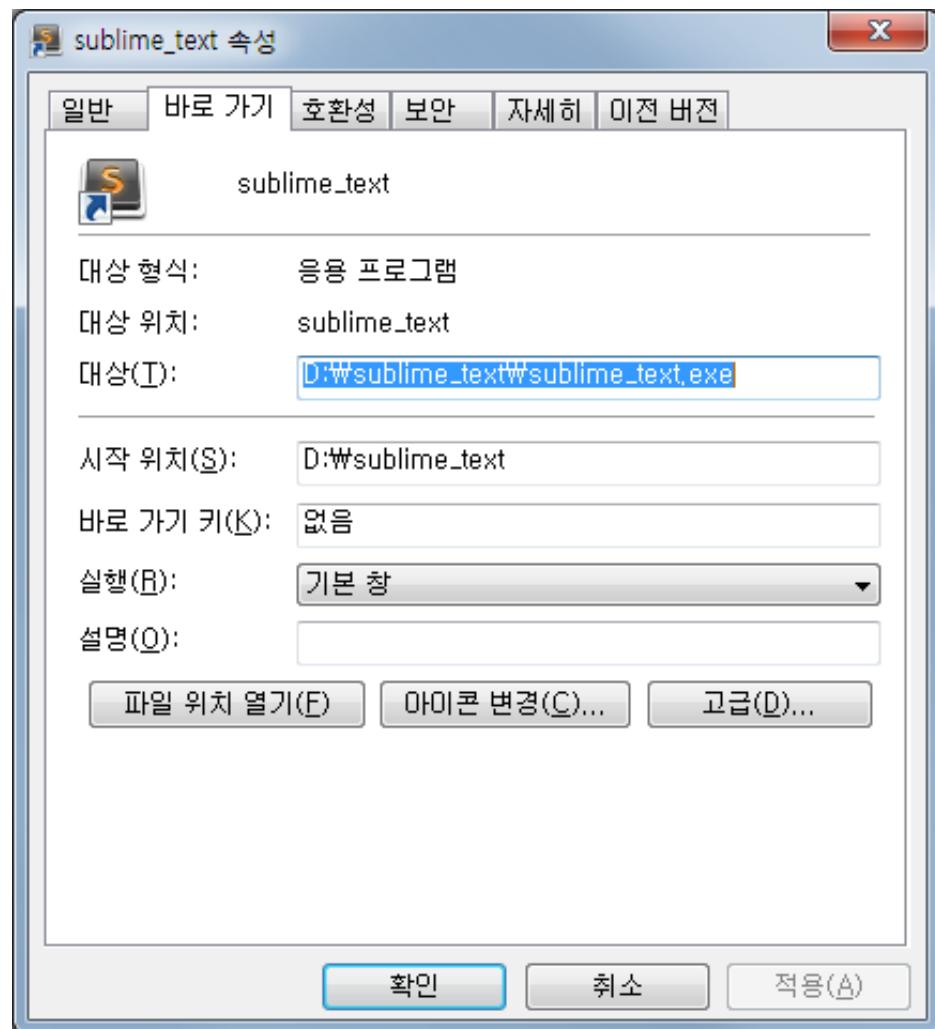
[생각해보기]



실행 시켜 보기

- 1) 대상에 있는 **text**전체를 복사
- 2) **Windows key + r**을 누르고
- 3) **cmd** 를 입력 후 엔터를 칠 것
- 4) 오른쪽 마우스를 클릭
- 5) “붙여넣기” 메뉴선택 한 후
- 6) 텍스트를 입력한 후 Enter
- 7) 해당 프로그램이 실행됨

[생각해보기]

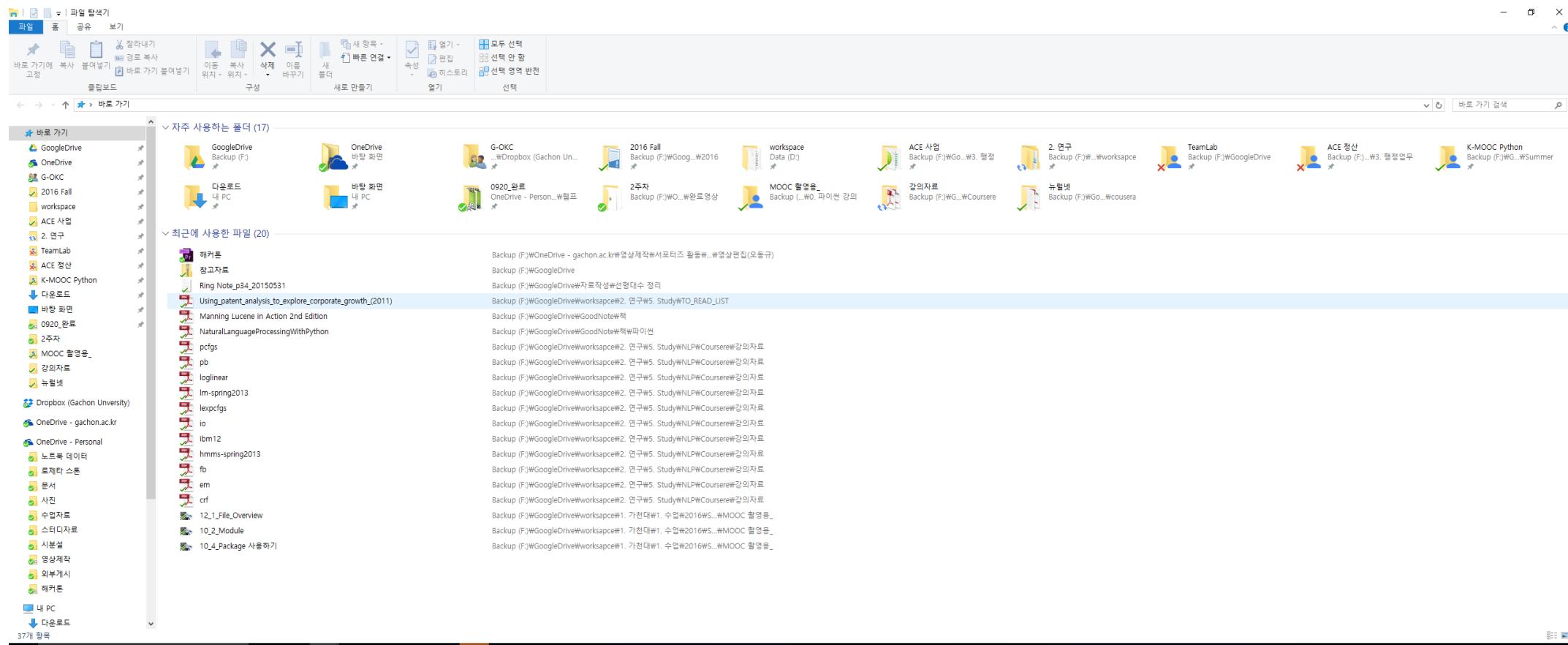


실행 시켜 보기

- 1) 대상에 있는 **text** 전체로는
 - 2) **Windows key** + **S**를 누르는 것은 실제로는
 - 3) **cmd** 를 누르는 것은 명령하는 것
 - 4) 아이콘을 누르는 것은 실행하는 것
 - 5) 파일의 “**실행**” 메뉴선택 한 후
 - 6) 터미널을 입력한 후 Enter
 - 7) 해당 프로그램이 실행됨
- 아이콘을 누르는 것은 실행하는 것**

파일의 이해

파일은 파일을 담고 있는 디렉토리와 파일로 나눌 수 있음



파일의 이해

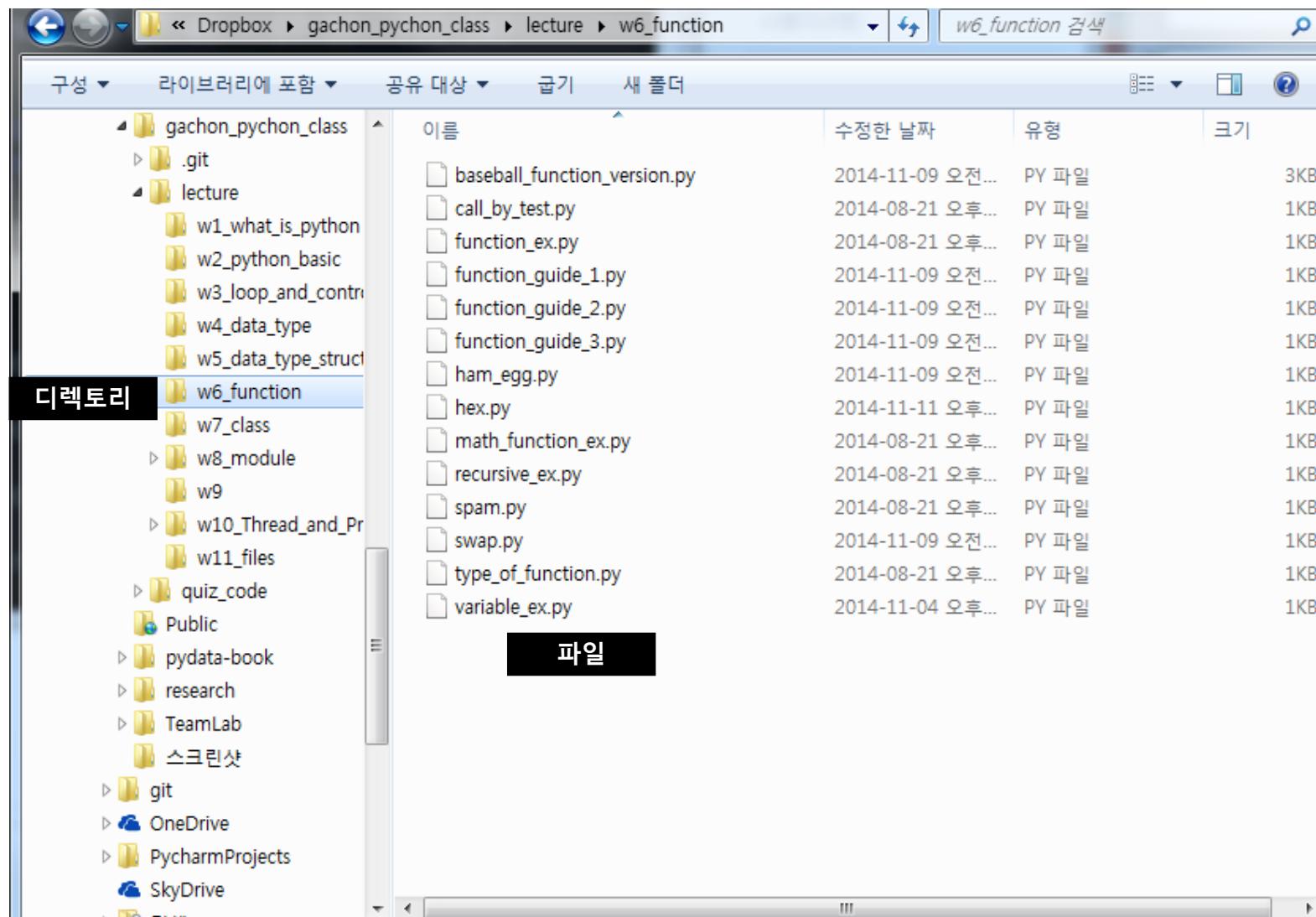
디렉토리 (Directory)

- 폴더 또는 디렉토리로 불림
- 파일과 다른 디렉토리를 포함할 수 있음

파일 (File)

- 컴퓨터에서 정보를 저장하는 논리적인 단위 ([wikipedia](#))
- 파일은 파일명과 확장자로 식별됨 (예: hello.py)
- 실행, 쓰기, 읽기 등을 할 수 있음

파일의 구조(Windows)



파일의 구조(Mac OS)

```
./lecture/w10_Thread_and_Process: 디렉토리
합계 52
-rw-r--r-- 1 root root 1486 2014-11-09 03:28 BubbleSort.class
-rw-r--r-- 1 root root 1313 2014-11-09 03:28 BubbleSort.java
drwxr-xr-x 2 root root 4096 2014-11-12 18:06 bubble_sort_comparison 파일
-rw-r--r-- 1 root root 642 2014-11-09 03:28 bubble_sort_gevent.py
-rw-r--r-- 1 root root 549 2014-11-09 03:28 bubble_sort_non_thread.py
-rw-r--r-- 1 root root 768 2014-11-09 03:28 bubble_sort_process.py
-rw-r--r-- 1 root root 1678 2014-11-09 03:28 bubble_sort_process.pyc
-rw-r--r-- 1 root root 636 2014-11-09 03:28 bubble_sort_thread.py
-rw-r--r-- 1 root root 319 2014-11-09 03:28 process_basic.ex.py
-rw-r--r-- 1 root root 781 2014-11-09 03:28 process_basic.ex.pyc
-rw-r--r-- 1 root root 270 2014-11-09 03:28 thread_baisc_ex.py
-rw-r--r-- 1 root root 442 2014-11-09 03:28 thread_baisc_ex_2.py
-rw-r--r-- 1 root root 130 2014-11-09 03:28 thread_basic_code.py

./lecture/w10_Thread_and_Process/bubble_sort_comparison:
합계 8
-rw-r--r-- 1 root root 585 2014-11-09 03:28 code_a.py
-rw-r--r-- 1 root root 579 2014-11-09 03:28 code_b.py

./lecture/w11_files:
합계 0
```

파일의 종류

Binary 파일	Text 파일
<ul style="list-style-type: none">- 컴퓨터만 이해할 수 있는 형태인 이진(법)형식으로 저장된 파일- 일반적으로 메모장으로 열면 내용이 깨져 보임 (메모장 해설 불가)- 엑셀파일, 워드 파일 등등	<ul style="list-style-type: none">- 인간도 이해할 수 있는 형태인 문자열 형식으로 저장된 파일- 메모장으로 열면 내용 확인 가능- 메모장에 저장된 파일, HTML 파일, 파이썬 코드 파일 등

- 컴퓨터는 Text 파일을 처리하기 위해 Binary 파일로 변환시킴 (예: pyc 파일)
- 모든 Text 파일도 실제는 Binary 파일, ASCII/Unicode 문자열 집합으로 저장되어 사람이 읽을 수 있음

파일의 종류

ASCII CODE TABLE

10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자
0	0x00	NULL	22	0x16	STN	44	0x2C	,	66	0x42	B	88	0x58	X	110	0x6E	n
1	0x01	SOH	23	0x17	ETB	45	0x2D	-	67	0x43	C	89	0x59	Y	111	0x6F	o
2	0x02	STX	24	0x18	CAN	46	0x2E	.	68	0x44	D	90	0x5A	Z	112	0x70	p
3	0x03	ETX	25	0x19	EM	47	0x2F	/	69	0x45	E	91	0x5B	[113	0x71	q
4	0x04	EOT	26	0x1A	SUB	48	0x30	0	70	0x46	F	92	0x5C	₩	114	0x72	r
5	0x05	ENQ	27	0x1B	ESC	49	0x31	1	71	0x47	G	93	0x5D]	115	0x73	s
6	0x06	ACK	28	0x1C	FS	50	0x32	2	72	0x48	H	94	0x5E	^	116	0x74	t
7	0x07	BEL	29	0x1D	GS	51	0x33	3	73	0x49	I	95	0x5F	-	117	0x75	u
8	0x08	BS	30	0x1E	RS	52	0x34	4	74	0x4A	J	96	0x60	`	118	0x76	v
9	0x09	HT	31	0x1F	US	53	0x35	5	75	0x4B	K	97	0x61	a	119	0x77	w
10	0x0A	₩n	32	0x20	SP	54	0x36	6	76	0x4C	L	98	0x62	b	120	0x78	x
11	0x0B	VT	33	0x21	!	55	0x37	7	77	0x4D	M	99	0x63	c	121	0x79	y
12	0x0C	FF	34	0x22	"	56	0x38	8	78	0x4E	N	100	0x64	d	122	0x7A	z
13	0x0D	₩r	35	0x23	#	57	0x39	9	79	0x4F	O	101	0x65	e	123	0x7B	{
14	0x0E	SO	36	0x24	\$	58	0x3A	:	80	0x50	P	102	0x66	f	124	0x7C	
15	0x0F	SI	37	0x25	%	59	0x3B	;	81	0x51	Q	103	0x67	g	125	0x7D	}
16	0x10	DLE	38	0x26	&	60	0x3C	<	82	0x52	R	104	0x68	h	126	0x7E	~
17	0x11	DC1	39	0x27	'	61	0x3D	=	83	0x53	S	105	0x69	i	127	0x7F	DEL
18	0x12	DC2	40	0x28	(62	0x3E	>	84	0x54	T	106	0x6A	j			
19	0x13	DC3	41	0x29)	63	0x3F	?	85	0x55	U	107	0x6B	k			
20	0x14	DC4	42	0x2A	*	64	0x40	@	86	0x56	V	108	0x6C	l			
21	0x15	NAK	43	0x2B	+	65	0x41	A	87	0x57	W	109	0x6D	m			

출처:
<http://sexy.pe.kr>

TEAMLAB

Human knowledge belongs to the world.

File Handling

File

최성철 교수
Director of TEAMLAB

01100
00110

파이썬의 File I/O

파이썬은 파일 처리를 위해 “open” 키워드를 사용함

```
f = open("<파일이름>", "접근 모드")  
f.close()
```

파일열기모드	설명
r	읽기모드 - 파일을 읽기만 할 때 사용
w	쓰기모드 - 파일에 내용을 쓸 때 사용
a	추가모드 - 파일의 마지막에 새로운 내용을 추가 시킬 때 사용

파일 읽기

파이썬의 File Read

`read()` txt 파일 안에 있는 내용을 문자열로 반환

```
f = open("i_have_a_dream.txt", "r")
contents = f.read()
print(contents)
f.close()
```

대상파일이 같은 폴더에 있을 경우

`with` 구문과 함께 사용하기

```
with open("i_have_a_dream.txt", "r") as my_file:
    contents = my_file.read()
    print(type(contents), contents)
```

파이썬의 File Read

한 줄씩 읽어 List Type으로 반환함

```
with open("i_have_a_dream.txt", "r") as my_file:  
    content_list = my_file.readlines() #파일 전체를 list로 반환  
    print(type(content_list)) #Type 확인  
    print(content_list) #리스트 값 출력
```

파이썬의 File Read

실행 시마다 한 줄씩 읽어오기

```
with open("i_have_a_dream.txt", "r") as my_file:  
    i = 0  
    while 1:  
        line = my_file.readline()  
        if not line:  
            break  
        print(str(i) + " == " + line.replace("\n", "")) #한줄씩 값 출력  
        i = i + 1
```

파이썬의 File Read

단어 통계 정보 산출

```
with open("i_have_a_dream.txt", "r") as my_file:  
    contents = my_file.read()  
    word_list = contents.split(" ")  
    #빈칸 기준으로 단어를 분리 리스트  
    line_list = contents.split("\n")  
    #한줄 씩 분리하여 리스트  
  
    print("Total Number of Characters :", len(contents))  
    print("Total Number of Words:", len(word_list))  
    print("Total Number of Lines :", len(line_list))
```

파일 쓰기

파이썬의 File Write

mode는 “w”, encoding=“utf8”

```
f = open("count_log.txt", 'w', encoding="utf8")
for i in range(1, 11):
    data = "%d번째 줄입니다.\n" % i
    f.write(data)
f.close()
```

mode는 “a”는 추가 모드

```
with open("count_log.txt", 'a', encoding="utf8") as f:
    for i in range(1, 11):
        data = "%d번째 줄입니다.\n" % i
        f.write(data)
```

파이썬의 Directory 만들기

os 모듈을 사용하여 Directory 다루기

```
import os  
os.mkdir("log")
```

디렉토리가 있는지 확인하기

```
if not os.path.isdir("log"):  
    os.mkdir("log")
```

Log 파일 생성하기

1) 디렉토리가 있는지, 2) 파일이 있는지 확인 후

```
import os
if not os.path.isdir("log"):
    os.mkdir("log")
if not os.path.exists("log/count_log.txt"):
    f = open("log/count_log.txt", 'w', encoding="utf8")
    f.write("기록이 시작됩니다\n")
    f.close()

with open("log/count_log.txt", 'a', encoding="utf8") as f:
    import random, datetime
    for i in range(1, 11):
        stamp = str(datetime.datetime.now())
        value = random.random() * 1000000
        log_line = stamp + "wt" + str(value) + "값이 생성되었습니다" + "\n"
        f.write(log_line)
```

python pickle

Pickle

- 파이썬의 객체를 영속화(persistence)하는 built-in 객체
- 데이터, object 등 실행중 정보를 저장 → 불러와서 사용
- 저장해야하는 정보, 계산 결과(모델) 등 활용이 많음

```
import pickle

f = open("list.pickle", "wb")
test = [1, 2, 3, 4, 5]
pickle.dump(test, f)
f.close()

f = open("list.pickle", "rb")
test_pickle = pickle.load(f)
print(test_pickle)
f.close()
```

Pickle

```
import pickle

class Mutltiplay(object):
    def __init__(self, multiplier):
        self.multiplier = multiplier

    def multiply(self, number):
        return number * self.multiplier

mulpily = Mutltiplay(5)
mulpily.multiply(10)

f = open("multiply_object.pickle", "wb")
pickle.dump(mulpily, f)
f.close()

f = open("multiply_object.pickle", "rb")
multiply_pickle = pickle.load(f)
multiply_pickle.multiply(5)
```

TEAMLAB

Human knowledge belongs to the world.

Lab – News Categorization

File

최성철 교수
Director of TEAMLAB

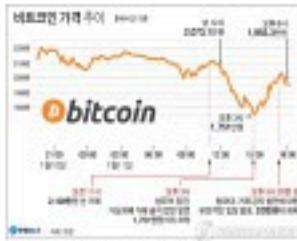
01100
00110

**비슷한 뉴스를
어떻게 선정할까?**

가상화폐 광풍

거래소 폐쇄 방침 '흔선'에 비트코인 가격 '롤러코스터'

연합뉴스 · ○ 203



비트코인 '거래소 폐지 방침'에 우르르...한...

연합뉴스

정부 가상화폐 때리기에 靑 몰려간 투자자... 한

연합뉴스

비트코인 광맥 끊기나..중국 채굴 전면 금지

머니투데이 · ○ 327



오라클 웹로직 서버, 암호화폐 채굴에 쓰였다 지디넷코리아

"경찰에 국세청까지" 규제공포에 암호화폐 ... 뉴스1

"비트코인은 인생의 동아줄"

2030은 왜?

조선일보 · ○ 441



4차 산업혁명..국민 관심사는 암호화폐? 지디넷코리아

페이스북 부사장 "한국투자 확대..중소기업과 협력 강화"

연합뉴스 · ○ 7



외국계IT기업 '국내 법적 대리인' 의무화 매일경제

'망 무임승차' 논란 페이스북 "사용료 문제" ... 연합뉴스

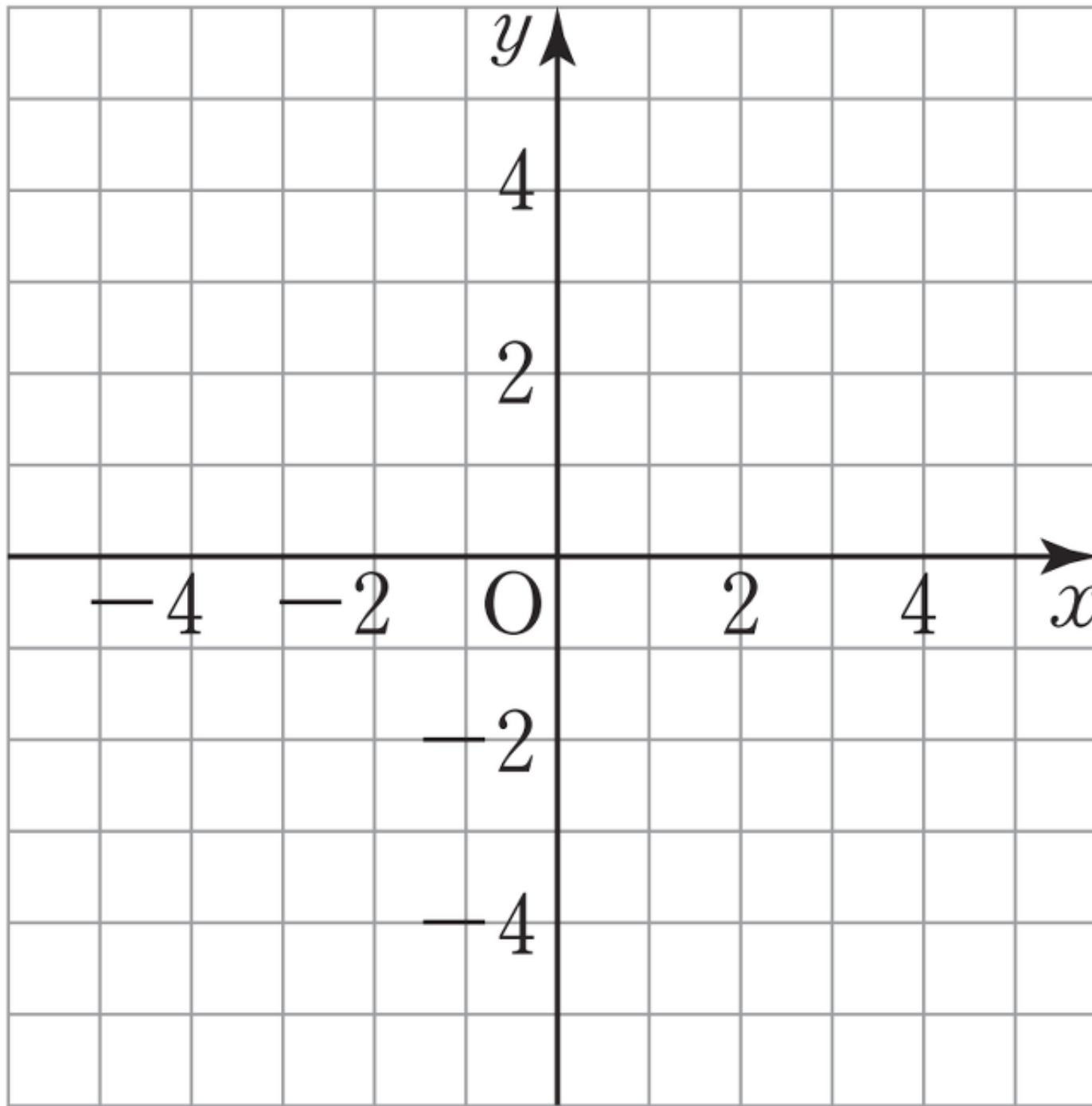
<http://media.daum.net/digital/>

**컴퓨터는 문자를
그대로 이해하지 못함**

문자 → 숫자

**숫자로 유사하다는
어떻게 표현할까?**

유사하다=가깝다



문자 → 숫자 → Vector

문자를 Vector로 – One-hot Encoding

- 하나의 단어를 Vector의 Index로 인식, 단어 존재시 1 없으면 0

The diagram illustrates the one-hot encoding of four words: Rome, Paris, Italy, and France. Each word is associated with a specific index in a vector of length V. The word 'Rome' is at index 0, 'Paris' is at index 1, 'Italy' is at index 2, and 'France' is at index 3. Arrows point from each word to its corresponding index in the vectors below.

Rome	=	[1, 0, 0, 0, 0, 0, ..., 0]
Paris	=	[0, 1, 0, 0, 0, 0, ..., 0]
Italy	=	[0, 0, 1, 0, 0, 0, ..., 0]
France	=	[0, 0, 0, 1, 0, 0, ..., 0]

Bag of words

- 단어별로 인덱스를 부여해서, 한 문장(또는 문서)의 단어의 개수를 Vector로 표현

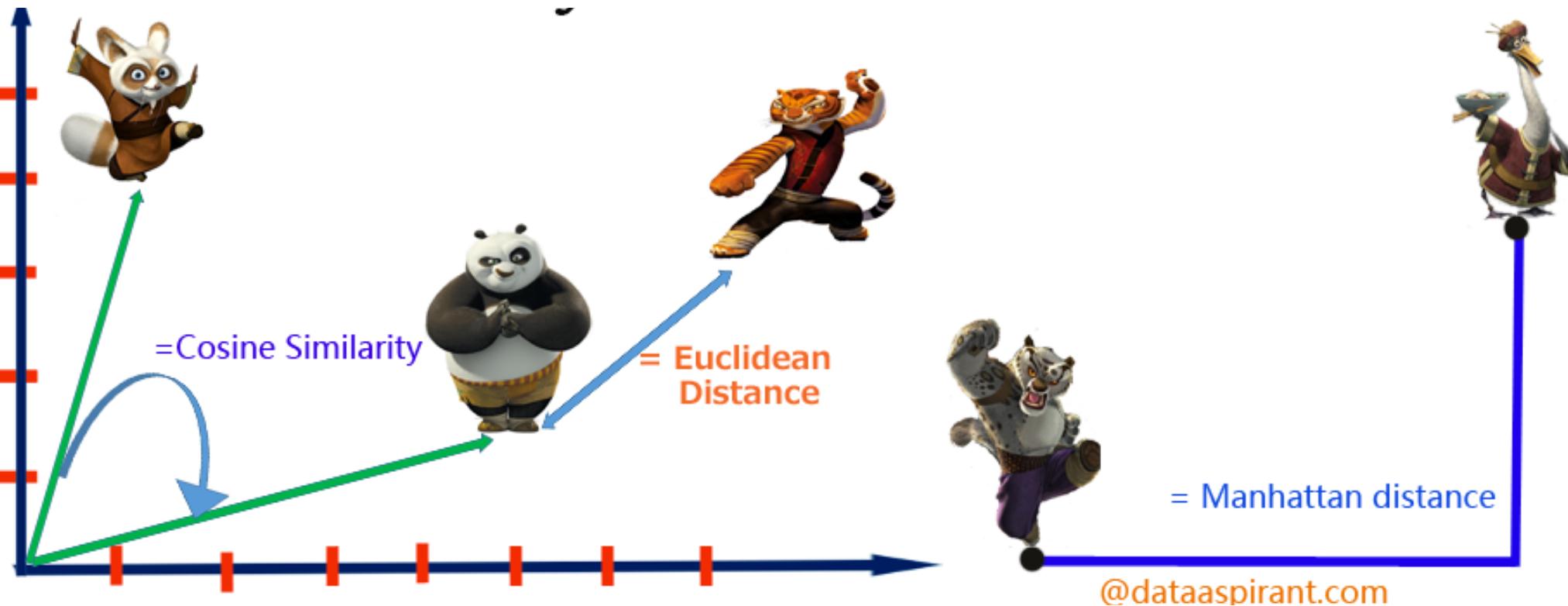
the dog is on the table



그렇다면 유사성은?

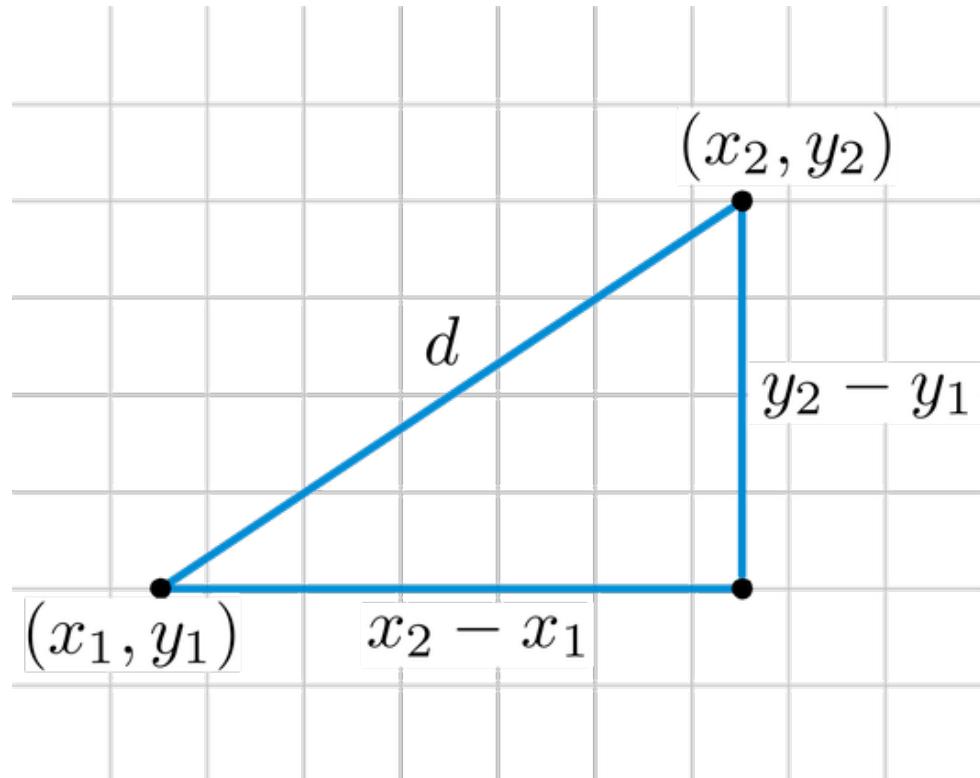
Distance measure

- 고등학교때 배운 2차원 평면상 거리측정 방법들



Euclidian distance

- 피타고拉斯 정리, 두 점 사이의 직선의 거리



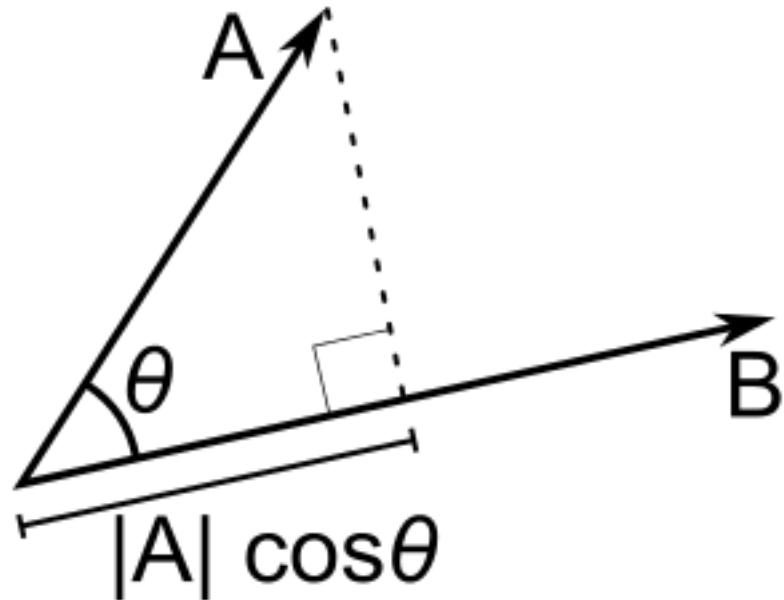
<https://goo.gl/cVmeKr>

$$d(a, b) = \sqrt{\sum_i^n (a_i - b_i)^2}$$

$[a_1 \quad a_2 \quad a_3 \quad \dots \quad a_n]$
 $[b_1 \quad b_2 \quad b_3 \quad \dots \quad b_n]$

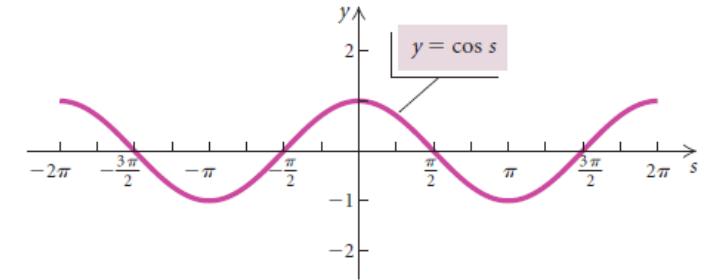
Cosine distance

- 두 점 사이의 각도



$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=0}^n A_i \times B_i}{\sqrt{\sum_{i=0}^n (A_i)^2} \times \sqrt{\sum_{i=0}^n (B_i)^2}}$$

$$\therefore A \cdot B = AB \cos(\theta)$$



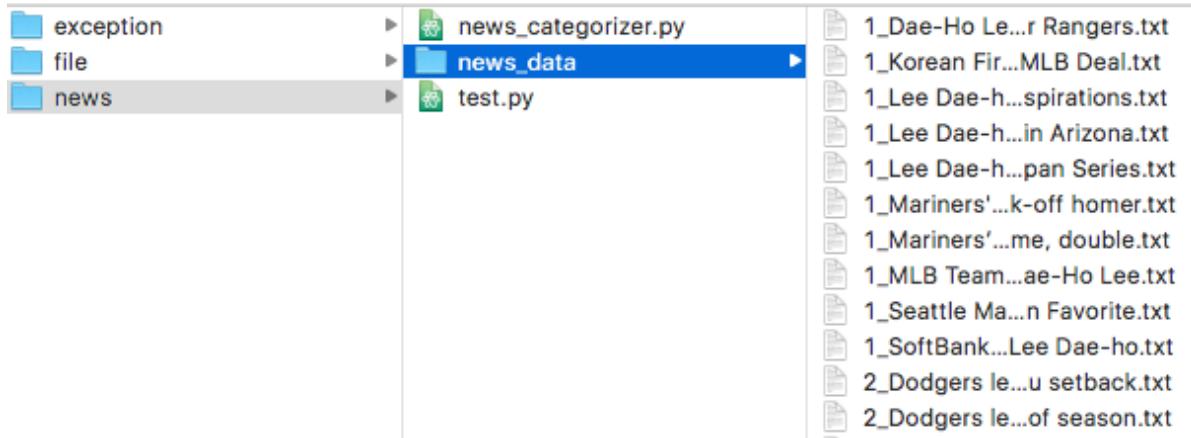
Cosine distance

- Why cosine similarity? Count < Direction
- Love,hate (5,0), (5,4), (4,0) , 어느점이 가장 가까운가

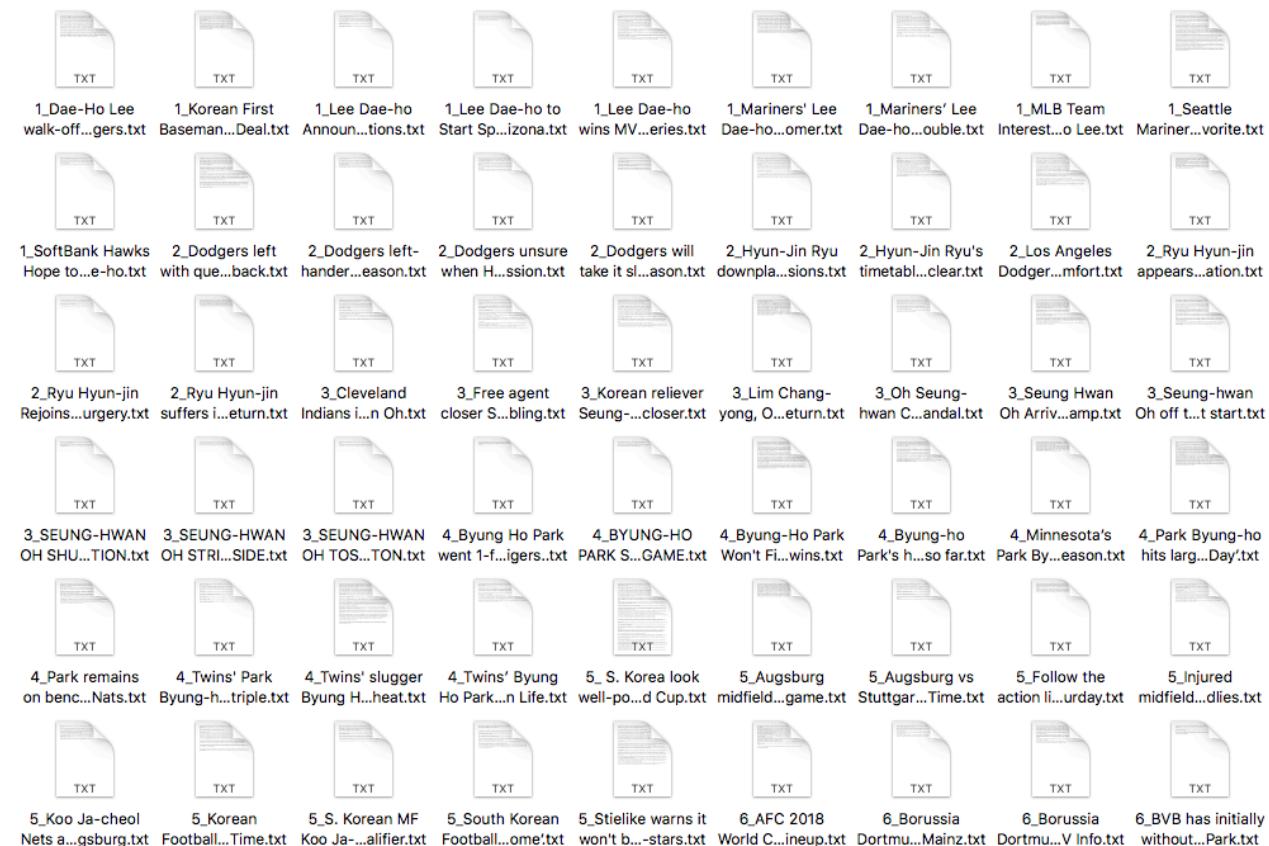
Codes

Data set

- 축구와 야구 선수들의 영문 기사를 분류해보자!



1,2,3,4 야구
5,6,7,8 축구



Process

- 파일을 불러오기
- 파일을 읽어서 단어사전 (corpus) 만들기
- 단어별로 Index 만들기
- 만들어진 인덱스로 문서별로 Bag of words vector 생성
- 비교하고자 하는 문서 비교하기
- 얼마나 맞는지 측정하기

파일 불러오기

```
def get_file_list(dir_name):
    return os.listdir(dir_name)

if __name__ == "__main__":
    dir_name = "news_data"
    file_list = get_file_list(dir_name)
    file_list = [os.path.join(dir_name, file_name) for file_name in file_list]
```

파일별로 내용읽기

```
def get_contents(file_list):
    y_class = []
    X_text = []
    class_dict = {
        1: "0", 2: "0", 3:"0", 4:"0", 5:"1", 6:"1", 7:"1", 8:"1"}

    for file_name in file_list:
        try:
            f = open(file_name, "r", encoding="cp949")
            category = int(file_name.split(os.sep)[1].split("_")[0])
            y_class.append(class_dict[category])
            X_text.append(f.read())
            f.close()
        except UnicodeDecodeError as e:
            print(e)
            print(file_name)
    return X_text, y_class
```

Corpus 만들기 + 단어별 index 생성하기

```
def get_cleaned_text(text):    의미없는 문장보호 등을 제거하기
    import re
    text = re.sub('\W+', '', text.lower() )
    return text
```

```
def get_corpus_dict(text):
    text = [sentence.split() for sentence in text]
    cleaned_words = [get_cleaned_text(word) for words in text for word in words]
```

```
from collections import OrderedDict
corpus_dict = OrderedDict()
for i, v in enumerate(set(cleaned_words)):
    corpus_dict[v] = i
return corpus_dict
```

문서별로 Bag of words vector 생성

```
def get_count_vector(text, corpus):
    text = [sentence.split() for sentence in text]
    word_number_list = [[corpus[get_cleaned_text(word)]] for word in words]
for words in text]
    X_vector = [[0 for _ in range(len(corpus))] for x in range(len(text))]

    for i, text in enumerate(word_number_list):
        for word_number in text:
            X_vector[i][word_number] += 1
return X_vector
```

비교하기

```
import math
def get_cosine_similarity(v1,v2):
    "compute cosine similarity of v1 to v2: (v1 dot
v2)/{||v1||*||v2||}"
    sumxx, sumxy, sumyy = 0, 0, 0
    for i in range(len(v1)):
        x = v1[i]; y = v2[i]
        sumxx += x*x
        sumyy += y*y
        sumxy += x*y
    return sumxy/math.sqrt(sumxx*sumyy)
```

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=0}^n A_i \times B_i}{\sqrt{\sum_{i=0}^n (A_i)^2} \times \sqrt{\sum_{i=0}^n (B_i)^2}}$$

비교결과 정리하기

```
def get_similarity_score(X_vector, source):
    source_vector = X_vector[source]
    similarity_list = []
    for target_vector in X_vector:
        similarity_list.append(
            get_cosine_similarity(source_vector, target_vector))
    return similarity_list

def get_top_n_similarity_news(similarity_score, n):
    import operator
    x = {i:v for i, v in enumerate(similarity_score)}
    sorted_x = sorted(x.items(), key=operator.itemgetter(1))

    return list(reversed(sorted_x))[1:n+1]
```

성능 측정하기

```
def get_accuracy(similarity_list, y_class, source_news):
    source_class = y_class[source_news]

    return sum([source_class == y_class[i[0]] for i in similarity_list]) /
len(similarity_list)

for i in range(80):
    source_number = i

    similarity_score = get_similarity_score(X_vector, source_number)
    similarity_news = get_top_n_similarity_news(similarity_score, 10)
    accuracy_score = get_accuracy(similarity_news, y_class, source_number)
    result.append(accuracy_score)
print(sum(result) / 80)
```

TEAMLAB

Human knowledge belongs to the world.