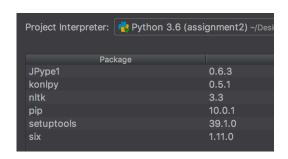
인공지능 #2

영화 리뷰 긍정 부정 판별하기

#### 1. Environment

konlpy, nltk 로 형태소 분석 사용



nltk.download() 로 nltk 관련 형태소 tag 다운



### 2. 코드 설명

- 더 좋은 결과를 얻기 전 가장 기본 코드

### 2-1 파일 읽기

line 별로 읽는다 id , document, label 로 분류한다 negative, positive 전체 확률위해 cnt 를 1씩 더한다

train으로 단어에 대한 확률을 update한다

#### 2-2 훈련시키기

```
# https://github.com/chandong83/python_hang
def is_hangul(text):
    return len(re.findall(u'[\u3130-\u318F\

# state : 0 negative / 1 positive
def train(text, state):
    if is_hangul(text):
        train_hangul(text, state)
    else:
        train_not_hangul(text, state)
```

github.com/chandong83/python\_hangul\_check\_function 한글 문장인지 체크하는 regular expression

한글(한글 + 영어) 문장이라면 한글 train으로 영어 문장이라면 영어 train으로 훈련시킨다

### 2-3 훈련시키기 - 한글

```
t = Twitter()

def train_hangul(text, state):
    pair_word_tag_list = t.pos(text)

for _word, tag in pair_word_tag_list:
    if _word not in word_ko_dict:
        word_ko_dict[_word].append(0)
        word_ko_dict[_word].append(0)

if state == '0':
        word_ko_dict[_word][0] += 1
    else:
    word_ko_dict[_word][1] += 1
```

아직 tag값은 사용하지 않고 work token만 가지고 word\_ko\_dictionary 에 긍정값인지 부정값인지 확률을 측정하기 위 해 카운팅

### 2-4 훈련시키기 - 영어

```
def train_not_hangul(text, state):
    text = text.lower()
    words = nltk.word_tokenize(text)
    pair_word_tag_list = nltk.pos_tag(words)

for _word, tag in pair_word_tag_list:
    if _word not in word_not_ko_dict:
        word_not_ko_dict[_word].append(0)
    word_not_ko_dict[_word].append(0)

if state == '0':
    word_not_ko_dict[_word][0] += 1
    else:
    word_not_ko_dict[_word][1] += 1
```

아직 tag값은 사용하지 않고 work token만 가지고 word\_not\_ko\_dictionary 에 긍정값인지 부정값인지 확률을 측정 하기 위해 카운팅

# 2-5 분류시키기

```
print("classify..")
  result_cnt = 0
  result_correct = 0
  with open("ratings_valid.txt", encoding="utf-8") as f:
    for line in f:
        if line[-1] is "\n":
            line = line[:-1]
        split = line.split("\t")
        id_movie = split[0]
        document_movie = split[1]
        label_movie = split[2]

    result_cnt += 1
    if label_movie == classify(document_movie):
        result_correct += 1
        print("OK : ", document_movie, label_movie)
    else:
        print("NO : ", document_movie, label_movie)

print(result_cnt)
print(result_correct)
```

한줄 씩 읽으면서 classify로 긍정인지 부정인지 판단한다 아래 소스는 valid를 check하는 것으로 정확도를 측정하기 위한 코드

# 2-6 분류시키기 2

한글 인지 아닌지는 dictionary를 check하는데만 사용

```
p(pos|words) 랑 p(neg|words)를 위해
p(pos) * p(word1|pos) * p(word2|neg) ....
p(neg) * p(word1|neg) * p(word2|neg) ....
```

를 위해서 log를 씌운 값을 더해서 pos가 큰지 neg가 큰지 return한다

## 3 실행결과

## 3-1 형태소에 따로 기능을 추가하지 않고 token화만 한 후 분류한 결과

```
OK : 금니스의 급도로마다 V

OK : 그레이스의 겉모습만 보던 사람들에게 그녀의 마음을 보여준 영화라고 생각함..

OK : 범무 0

OK : 정말 제밌게 본 컬트영화 1

NO : 같은시나리오에 다른 배우들이 연기했음 어땠을까? 1

OK : 마이크리의 영화는 항상 조용하면서 강력하다 1

OK : 대 인권의 영화 1

OK : 해 인권의 영화 1

OK : 최 생동빛간한데 소재가 재미있었네요 다음작품 한번 기대해봅니다. 1

OK : 최 생동빛간한데 소재가 재미있었네요 다음작품 한번 기대해봅니다. 1

OK : 처음으로 영화판에서 잠 0

OK : 보는 내내 울었다 너무 슬프고 감동적이야 ㅠㅠ 1

OK : 이ト~ 담주까지 어케 기다려요!! 극분, 대사 대박~~ 검사, 청창, 주인공4명

OK : 하품이 나오는 출작 0

18091
```

# 앞으로 할 것을 위한 참고자료

## - 한글

```
twitter-korean-testle normalization, tokenization, stemming, phrase extraction 이렇게 되기지 기능을 지원합니다.
경구화 normalization (입니덕국 = > 입니다 국국, 사용해 - > 사업해)

한 국어를 처리하는 예시입니다 국국 - > 한국어를 처리하는 예시입니다 국국
문력 tokenization

한 국어를 처리하는 에시입니다 국국 -> 한국어Noun, $1000, 처리Noun, 하는Verb, 에시Noun, 입Adjective, 니디Eomi 국국
KoreanParticle
이근화 stemming (입니다 -> 이다)

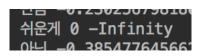
한국어를 처리하는 예시입니다 국국 -> 한국어Noun, $1000, 처리Noun, 하다Verb, 에시Noun, 이디Adjective, 국국KoreanParticle
이구 3을 phrase extraction

한 국어를 처리하는 예시입니다 국국 -> 한국어, 처리, 예시, 처리하는 예시
Introductory Presentation: Google Slides
```

### - 영어

OS tag	description		example
CC	Coordinating conjunction	등위 접속사	for, and, nor, but, or, yet, so
CD	cardinal number	기수	one, two, 35, 10, etc.
DT	Determiner	한정사	명사 앞에 나와서 명사를 수식하는 역할 a, that, any, all, this, etc.
EX	Existential there		존재의 there there is an apple
PW	Foreign word	외래어	dong, alam 등 트륏에서 자주 발생하는 misspelled word들이 많이
IN	preposition	전치사	because, ago, under, within, once, etc.
-1,	or subordinating conjunction	중속접속사	
JJ	adjective	현용사	sick, cheap, great, etc.
JJR	adjective , comparative	형용사, 비교급	better, more, happier, etc.
JJS	adjective, superlative	형용사, 최상급	best, highest, most, etc.
LS	List Item Marker		i, ii, iii, iv, etc.
MD	Modal	조동사	can, could, would, etc.
NN	Noun, singular or mass	명사	show, food, girl, etc.
NNP	proper noun, singular	고유명사(단수)	dad, mart, 요일, 퓔, etc.
NNPS	proper noun, plural	고유명사(복수)	
NNS	noun, plural	명사(복수)	
PDT	Predeterminer	전치한정사	such
POS	Possessive Ending		Nouns ending in's
PRP	Personal Pronoun	인칭대명사	you, he, I
PRP\$	Possessive Pronoun		my, your, mine
RB	adverb	부사	대부분 -ly로 끝남. Even, totally, personally
RBR	adverb, comparative		hotter, sooner
RBS	adverb, superlative		most, hardest
RP	particle	불변화사	up, on turn it on
SYM	symbol		used for mathematical, scientific, or technical symbols
TO	to		
UH	interjection	감탄사	oh, uh, well, yes
VB	verb base form	일반동사	
VBD		과거동사	got
VBG		동맹사, 현재분시	being
VBN		과거분사	enjoyed
VBP	non-3rd person singular presen		come, feel
VBZ	71,50	3인칭 단수	comes, feels
WDT	wh-determiner	-	which, that(관계대명사로 쓰였을때)
WP	wh-pronoun	관계대명사	what, who, whom
WP\$	Possessive wh-pronoun		whoever, who, what
WRB	wh-adverb		how, where, why, wherever

- 4 추가 작업
- 1. Infinity 가 되는 것을 막기 위해 양측에 threshold 값 추가 8079 -> 8441



-0.53924 -3.4057760 10001 8441

2. 한글이 아닌것 training에 stemmer 추가 8441 -> 8442

-3.40577607 10001 8442

```
for w in words:
    if w not in stop_words:
        words_filtered.append(stemmer.stem(w))
pair_word_tag_list = nltk.pos_tag(words_filtered)
```

- 3. 한글에 training에 stemmer 추가 8442 -> 7856
- 4. 한글 classify에도 stemmer추가 7856 -> 8337

```
train_hangul(text, state):
  pair_word_tag_list = t.pos(text, norm=True, stem=True)
```

5. 한글 training 에 classify에 Suffix, Josa, Punctuation 제거 8337 -> 8336

```
ignore_ko_tag = ['Suffix', 'Josa', 'Punctuation']
```

6. 한글 문장이 뒤쪽에 영향을 많이 받는 것 같아서 뒤쪽에 가중치를 둠 8336 -> 8315

```
pr _word, tag in pair_word_tag_list:
   power += (1 / length) * inc_power
```

7. 그 외에도 여러 변수에 대한 가중치를 추가해서 (1의 threshold값, 6의 power값 등) 8315 -> 8370대