2018.0.0

LDA 코드

김찬중, 양서윤, 정유진

© 2018. SNU Growth Hackers all rights reserved

Growth Hackers

CONTENTS

1. 데이터 전처리
2. 사전, corpus 생성
3. 모델 생성
4. 모델 평가
5. K 수 찾기

© 2018. SNU Growth Hackers all rights reserved

Import 목록

```
# Run in python console
import nltk

import re
import numpy as np
import pandas as pd
from pprint import pprint

# Gensim
import gensim
import gensim.corpora as corpora
from gensim.utils import simple_preprocess
from gensim.models import CoherenceModel

# spacy for lemmatization
import spacy

# Plotting tools
import pyLDAvis
import pyLDAvis.gensim  # don't skip this
import matplotlib.pyplot as plt


# Enable logging for gensim - optional
import logging

logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.ERROR)

import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings(action='ignore', category=UserWarning, module='gensim')

# NLTK Stop words
from nltk.corpus import stopwords
```

- Nltk

텍스트 전처리를 할 수 있는 페키지

- Gensim

LDA를 포함하고 있는 페키지

- Spacy

텍스트 전처리를 할 수 있는 페키지

- Warings

warning 나오는거 없애기

- Nltk vs spacy

http://yujuwon.tistory.com/entry/spaCy-사용하기

ⓒ 2018. SNU Growth Hackers all rights reserved

1. 데이터 전처리 – 데이터 불러오기

```
# Import Dataset
def to_table(filename):
    with open(filename, 'r') as apple:
        a = []
        for apps in apple.readlines():
            b = (apps.strip())
            b = b.strip('.,?!;-')
            a.append(b)
            line_list = []
            for line in a:
                line = line.strip('.,?!;-')
                line_list += line.split()
    return line_list

df = to_table('test_text(2).txt')
# print(df)
```

- Strip : 제거시키기
- Lower : 소문자로

```
['FOR', 'the', 'most', 'wild,', 'yet', 'most', 'homely', 'narrative', 'which', 'I', 'am', 'about', 'to', 'pe
uld', 'I', 'be', 'to', 'expect', 'it,', 'in', 'a', 'case', 'where', 'my', 'very', 'senses', 'reject', 'their
rely', 'do', 'I', 'not', 'dream.', 'But', 'to-morrow', 'I', 'die,', 'and', 'to-day', 'I', 'would', 'unburthe
'the', 'world,', 'plainly,', 'succinctly,', 'and', 'without', 'comment,', 'a', 'series', 'of', 'mere', 'hous
errified', '--', 'have', 'tortured', '--', 'have', 'destroyed', 'me.', 'Yet', 'I', 'will', 'not', 'attempt',
'but', 'Horror', '--', 'to', 'many', 'they', 'will', 'seem', 'less', 'terrible', 'than', 'barroques.', 'Here
'reduce', 'my', 'phantasm', 'to', 'the', 'common-place', '--', 'some', 'intellect', 'more', 'calm,', 'more',
'will', 'perceive,', 'in', 'the', 'circumstances', 'I', 'detail', 'with', 'awe,', 'nothing', 'more', 'than',
cts', 'From', 'my', 'infancy', 'I', 'was', 'noted', 'for', 'the', 'docility', 'and', 'humanity', 'of', 'my',
cuous', 'as', 'to', 'make', 'me', 'the', 'jest', 'of', 'my', 'companions.', 'I', 'was', 'especially', 'fond
'a', 'great', 'variety', 'of', 'pets.', 'With', 'these', 'I', 'spent', 'most', 'of', 'my', 'time,', 'and',
m.', 'This', 'peculiarity', 'of', 'character', 'grew', 'with', 'my', 'growth,', 'and,', 'in', 'my', 'manhoo
f', 'pleasure.', 'To', 'those', 'who', 'have', 'cherished', 'an', 'affection', 'for', 'a', 'faithful', 'and
f', 'explaining', 'the', 'nature', 'or', 'the', 'intensity', 'of', 'the', 'gratification', 'thus', 'derivab
ing', 'love', 'of', 'a', 'brute,', 'which', 'goes', 'directly', 'to', 'the', 'heart', 'of', 'him', 'who', 'h
'and', 'gossamer', 'fidelity', 'of', 'mere', 'Man', 'I', 'married', 'early,', 'and', 'was', 'happy', 'to',
y', 'own.', 'Observing', 'my', 'partiality', 'for', 'domestic', 'pets,', 'she', 'lost', 'no', 'opportunity',
ad', 'birds,', 'gold-fish,', 'a', 'fine', 'dog,', 'rabbits,', 'a', 'small', 'monkey,', 'and', 'a', 'cat', '
```

© 2018. SNU Growth Hackers all rights reserved

1. 데이터 전처리 – 형태소 나누기

```
tmp = []
for i in df:
    i = i.strip('.,?!;-').lower()
    texts = nltk.word_tokenize(i)
    tmp += nltk.pos_tag(texts)

print(tmp)
```

```
[('for', 'IN'), ('the', 'DT'), ('most', 'JJS'), ('wild', 'NN'), ('yet', 'RB'), ('most', 'JJS'), ('hc
t', 'IN'), ('to', 'TO'), ('pen', 'VB'), ('i', 'NN'), ('neither', 'DT'), ('expect', 'VB'), ('nor', 'C
D'), ('i', 'NN'), ('be', 'VB'), ('to', 'TO'), ('expect', 'VB'), ('it', 'PRP'), ('in', 'IN'), ('a', '
S'), ('reject', 'NN'), ('their', 'PRP$'), ('own', 'JJ'), ('evidence', 'NN'), ('yet', 'RB'), ('mad',
y', 'RB'), ('do', 'VB'), ('i', 'NN'), ('not', 'RB'), ('dream', 'NN'), ('but', 'CC'), ('to-morrow', '
d', 'MD'), ('unburthen', 'NN'), ('my', 'PRP$'), ('soul', 'NN'), ('my', 'PRP$'), ('immediate', 'NN'),
e', 'DT'), ('world', 'NN'), ('plainly', 'RB'), ('succinctly', 'RB'), ('and', 'CC'), ('without', 'IN'
sehold', 'NN'), ('events', 'NNS'), ('in', 'IN'), ('their', 'PRP$'), ('consequences', 'NNS'), ('these
tured', 'VBN'), ('have', 'VB'), ('destroyed', 'NN'), ('me', 'PRP'), ('yet', 'RB'), ('i', 'NN'), ('wi
'PRP'), ('to', 'TO'), ('me', 'PRP'), ('they', 'PRP'), ('have', 'VB'), ('presented', 'VBN'), ('little
P'), ('will', 'MD'), ('seem', 'NN'), ('less', 'RBR'), ('terrible', 'JJ'), ('than', 'IN'), ('barroque
('may', 'MD'), ('be', 'VB'), ('found', 'NN'), ('which', 'WDT'), ('will', 'MD'), ('reduce', 'VB'), ('
('some', 'DT'), ('intellect', 'NN'), ('more', 'RBR'), ('calm', 'NN'), ('more', 'RBR'), ('logical', '
N'), ('my', 'PRP$'), ('own', 'JJ'), ('which', 'WDT'), ('will', 'MD'), ('perceive', 'NN'), ('in', 'IN
N'), ('awe', 'NN'), ('nothing', 'NN'), ('more', 'RBR'), ('than', 'IN'), ('an', 'DT'), ('ordinary', '
'NNS'), ('and', 'CC'), ('effects', 'NNS'), ('from', 'IN'), ('my', 'PRP$'), ('infancy', 'NN'), ('i',
N'), ('and', 'CC'), ('humanity', 'NN'), ('of', 'IN'), ('my', 'PRP$'), ('disposition', 'NN'), ('my',
'RB'), ('so', 'RB'), ('conspicuous', 'JJ'), ('as', 'IN'), ('to', 'TO'), ('make', 'VB'), ('me', 'PRP'
```

- NLTK 부연설명

```
#영어 형태소 예시
# texts = nltk.word_tokenize("I am going to Seoul, Korea.")
# a=nltk.pos_tag(texts)
# print(a)
# http://sens.tistory.com/454
```

```
[('I', 'PRP'), ('am', 'VBP'), ('going', 'VBG'), ('to', 'TO'), ('Seoul', 'NNP'), (',', ','), ('Korea', 'NNP'),
('.', '.')]
```

© 2018. SNU Growth Hackers all rights reserved

Growth Hackers

NLTK 영어 형태소

POS 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, naturally, here, good
PRP$	possessive pronoun	my, his
RB	adverb	however, usually, naturally, here, good
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

- 출처 : http://sens.tistory.com/454

ⓒ 2018. SNU Growth Hackers all rights reserved

Growth Hackers

1. 데이터 전처리 - 필요한 형태소만 가져오기

```
# http://sens.tistory.com/454 참조, 형태소따라 나누기
def pos_extractor(parsed):

    noun_list = []
    adj_list = []
    verb_list = []
    nav_list = []
    adv_list = []

    for i in parsed:

        if i[1] in ['NN', 'NNS', 'NNP', 'NNPS']:
            noun_list.append(i[0])
            nav_list.append(i[0])
        elif i[1] in ['JJ', 'JJR', 'JJS']:
            adj_list.append(i[0])
            nav_list.append(i[0])
        elif i[1] in ['VB', 'VBD', 'VBG', 'VBN', 'VBP', 'VBZ']:
            verb_list.append(i[0])
            nav_list.append(i[0])
        elif i[1] == ['RB', 'RBR', 'RBS']:
            adv_list.append(i[0])
            nav_list.append(i[0])

        else:
            pass

    return [nav_list, noun_list, adj_list,
            verb_list]

print(pos_extractor(tmp))

data_words = list(pos_extractor(tmp)[0])
```

```
[['most', 'wild', 'most', 'narrative', 'i', 'am', 'pen', 'i', 'expect', 'solicit', 'belief', 'mad', 'i', 'be',
o', 'i', 'dream', 'to-morrow', 'i', 'die', 'to-day', 'i', 'unburthen', 'soul', 'immediate', 'purpose', 'is', 'p
'events', 'have', 'terrified', 'have', 'tortured', 'have', 'destroyed', 'i', 'attempt', 'expound', 'have', 'pres
ter', 'intellect', 'be', 'found', 'reduce', 'phantasm', 'common-place', 'intellect', 'calm', 'logical', 'excitab
rdinary', 'succession', 'natural', 'causes', 'effects', 'infancy', 'i', 'was', 'noted', 'docility', 'humanity',
'companions', 'i', 'was', 'fond', 'animals', 'was', 'indulged', 'parents', 'great', 'variety', 'pets', 'i', 'spe
'character', 'grew', 'growth', 'manhood', 'i', 'derived', 'principal', 'sources', 'pleasure', 'have', 'cherishe
e', 'explaining', 'nature', 'intensity', 'gratification', 'derivable', 'is', 'something', 'unselfish', 'self-sa
sion', 'test', 'paltry', 'friendship', 'gossamer', 'fidelity', 'man', 'i', 'married', 'was', 'happy', 'find', 'w
tic', 'pets', 'lost', 'opportunity', 'procuring', 'most', 'agreeable', 'kind', 'had', 'birds', 'gold-fish', 'fi
eautiful', 'animal', 'black', 'sagacious', 'astonishing', 'degree', 'speaking', 'intelligence', 'wife', 'heart'
n', 'ancient', 'popular', 'notion', 'regarded', 'black', 'cats', 'witches', 'disguise', 'was', 'serious', 'poin
o', 'was', 'cat', 'name', 'was', 'favorite', 'pet', 'playmate', 'i', 'fed', 'attended', 'wherever', 'i', 'went'
ndship', 'lasted', 'manner', 'several', 'years', 'general', 'temperament', 'character', 'instrumentality', 'fie
l', 'alteration', 'worse', 'i', 'grew', 'day', 'day', 'moody', 'irritable', 'feelings', 'others', 'i', 'suffere
onal', 'violence', 'pets', 'course', 'were', 'made', 'feel', 'change', 'disposition', 'i', 'neglected', 'ill-us
ng', 'i', 'made', 'scruple', 'maltreating', 'rabbits', 'monkey', 'dog', 'accident', 'affection', 'came', 'way',
coming', 'old', 'peevish', 'pluto', 'began', 'experience', 'effects', 'ill', 'temper', 'night', 'returning', 'h
```

- 명사, 부사, 형용사, 동사만 가져오기.

© 2018. SNU Growth Hackers all rights reserved

Growth Hackers

1. 데이터 전처리 - 불용단어 제거

```
#불용단어 준비
stop_words = stopwords.words('english')
stop_words.extend(['from', 'subject', 're', 'edu', 'use'])
```

```
def remove_stopwords(texts):
    return [[word for word in simple_preprocess(str(doc)) if word not in stop_words] for doc in texts]
```

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

```
# Remove Stop Words
data_words_nostops = remove_stopwords(data_words)
```

Simple_preprocess

Gensim에 있는 함수. str의 형태만 받는다. 너무 작거나 긴 단어 제외하고 lowercase로 변환시켜줌.

- https://www.geeksforgeeks.org/removing-stop-words-nltk-python/

ⓒ 2018. SNU Growth Hackers all rights reserved

1. 데이터 전처리 - Bigram, trigram 모델

- Bigram : 2연단어 ex) 'front_bumper', 'oil_leak'
- Trigram : 3연단어 ex) 'maryland_college_park'

```
# Build the bigram and trigram models
bigram = gensim.models.Phrases(data_words, min_count=5, threshold=100)  # higher threshold fewer phrases.
trigram = gensim.models.Phrases(bigram[data_words], threshold=100)

# Faster way to get a sentence clubbed as a trigram/bigram
bigram_mod = gensim.models.phrases.Phraser(bigram)
trigram_mod = gensim.models.phrases.Phraser(trigram)
```

ⓒ 2018. SNU Growth Hackers all rights reserved

1. 데이터 전처리 - bigram, trigram 모델

```
def make_bigrams(texts):
    return [bigram_mod[doc] for doc in texts]


def make_trigrams(texts):
    return [trigram_mod[bigram_mod[doc]] for doc in texts]


def lemmatization(texts, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV']):
    """https://spacy.io/api/annotation"""
    texts_out = []
    for sent in texts:
        doc = nlp(" ".join(sent))
        texts_out.append([token.lemma_ for token in doc if token.pos_ in allowed_postags])
    return texts_out
```

```
# print(data_words)
# print(trigram_mod[bigram_mod[data_words]])
```

ⓒ 2018. SNU Growth Hackers all rights reserved

1. 데이터 전처리 - bigram, trigram 모델

```
# Form Bigrams
data_words_bigrams = make_bigrams(data_words_nostops)

# Initialize spacy 'en' model, keeping only tagger component (for efficiency)
# python3 -m spacy download en (아나콘다 관리자권한에서 실행)
nlp = spacy.load('en', disable=['parser', 'ner'])
```

```
# Do lemmatization keeping only noun, adj, vb, adv
data_lemmatized = lemmatization(data_words_bigrams, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV'])
print(data_lemmatized)
```

```
[[], ['wild'], [], ['narrative'], [], [], ['pen'], [], ['expect'], ['solicit'], ['belief'], ['mad'], [], [], ['expe
[], [], ['dream'], ['morrow'], [], ['die'], ['day'], [], ['unburthen'], ['soul'], ['immediate'], ['purpose'], [], [
['consequence'], ['event'], [], ['terrify'], [], ['torture'], [], ['destroy'], [], ['attempt'], ['expound'], [], ['
rroque'], ['hereafter'], ['intellect'], [], ['find'], ['reduce'], ['phantasm'], ['common', 'place'], ['intellect'],
e'], [], ['detail'], ['awe'], ['nothing'], ['ordinary'], ['succession'], ['natural'], ['cause'], ['effect'], ['infa
enderness'], ['heart'], [], ['conspicuous'], ['make'], ['jest'], ['companion'], [], [], ['fond'], ['animal'], [], [
[], ['time'], [], ['happy'], ['feed'], ['caress'], ['peculiarity'], ['character'], ['grow'], ['growth'], ['manhood'
h'], ['affection'], ['faithful'], ['sagacious'], ['dog'], [], ['need'], [], ['trouble'], ['explain'], ['nature'], [
lfish'], ['self', 'sacrifice'], ['love'], ['brute'], ['go'], ['heart'], [], [], ['frequent'], ['occasion'], ['test'
['married'], [], ['happy'], ['find'], ['wife'], ['disposition'], ['uncongenial'], [], ['observe'], ['partiality'],
reeable'], ['kind'], [], ['bird'], ['gold', 'fish'], ['fine'], ['dog'], ['rabbit'], ['small'], ['monkey'], ['cat'],
ious'], ['astonishing'], ['degree'], ['speak'], ['intelligence'], ['wife'], ['heart'], [], ['little'], ['tincture']
['popular'], ['notion'], ['regard'], ['black'], ['cat'], ['witch'], ['disguise'], [], ['serious'], ['point'], [], [
o'], [], ['cat'], ['name'], [], ['favorite'], ['pet'], ['playmate'], [], ['feed'], ['attend'], ['wherever'], [], ['
t'], ['friendship'], ['last'], ['manner'], ['several'], ['year'], ['general'], ['temperament'], ['character'], ['in
s'], ['experienced'], ['radical'], ['alteration'], ['bad'], [], ['grow'], ['day'], ['day'], ['moody'], ['irritable'
e'], ['wife'], ['length'], [], ['offer'], ['personal'], ['violence'], ['pet'], ['course'], [], ['make'], ['feel'],
[], ['retain'], ['sufficient'], ['regard'], ['restrain'], ['maltreat'], [], ['make'], ['scruple'], ['maltreat'], ['
```

© 2018. SNU Growth Hackers all rights reserved

2. 사전, corpus 생성

```
# Create Dictionary
id2word = corpora.Dictionary(data_lemmatized)
# print(id2word)

# Create Corpus
texts = data_lemmatized
for i in texts:
    try:
        i == [] or ['₩n']
        texts.remove(i)
    except IndexError:
        pass

list_a = []
for i in texts:
    try:
        list_a.append(i[0])
    except IndexError:
        pass

# Term Document Frequency
corpus = [id2word.doc2bow(list_a) for text in texts]
# print(corpus)
# print(corpus[2]) # 일부가 출력되었음

# Human readable format of corpus (term-frequency)
[[(id2word[id], freq) for id, freq in cp] for cp in corpus[:1]]
```

```
Dictionary(91 unique tokens: ['unburthen', 'attempt', 'natural', 'growth', 'opportunity']...)
```

```
[['wild'], ['narrative'], ['solicit'], ['case'], ['reject'], ['evidence'], ['mad'], ['unburthen'], ['immediate'], ['
ttempt'], ['hereafter'], ['reduce'], ['common', 'place'], ['calm'], ['excitable'], ['perceive'], ['ordinary'], ['nat
e'], ['spend'], ['character'], ['growth'], ['principal'], ['pleasure'], ['cherish'], ['faithful'], ['need'], ['grati
y'], ['opportunity'], ['kind'], ['bird'], ['fine'], ['dog'], ['rabbit'], ['latter'], ['sagacious'], ['intelligence']
['black'], ['witch'], ['point'], ['mention'], ['pet'], ['attend'], ['street'], ['friendship'], ['several'], ['genera
eeling'], ['language'], ['pet'], ['change'], ['disposition'], ['ill', 'use'], ['sufficient'], ['restrain'], ['scrupl
['length'], ['pluto'], ['experience'], ['ill'], ['intoxicated'], ['town'], ['fancy'], ['fright'], ['inflict'], ['win
['pen', 'knife'], ['socket'], ['blush'], ['shudder'], ['pen'], ['reason'], ['return'], ['experienced'], ['horror'],
['suffer'], ['go'], ['expect'], ['approach'], ['old'], ['leave'], ['grieve'], ['love'], ['irritation'], ['overthrow'
['human'], ['indivisible'], ['faculty'], ['sentiment'], ['give'], ['character'], ['commit'], ['action'], ['reason'],
eness'], ['final'], ['overthrow'], ['unfathomable'], ['soul'], ['offer'], ['violence'], ['nature'], ['wrong'], ['urg
ing'], ['bitter'], ['remorse'], ['love'], ['give'], ['reason'], ['hang'], ['commit'], ['sin'], ['immortal'], ['soul'
['difficulty'], ['conflagration'], ['thenceforward'], ['weakness'], ['establish'], ['effect'], ['disaster'], ['atroc
t'], ['exception'], ['find'], ['bed'], ['great'], ['measure'], ['action'], ['fire'], ['fact'], ['attribute'], ['coll
ty'], ['approach'], ['graven'], ['relief'], ['surface'], ['gigantic'], ['give'], ['marvellous'], ['rope'], ['neck'],
['garden'], ['crowd'], ['animal'], ['cut'], ['tree'], ['open'], ['chamber'], ['do'], ['view'], ['sleep'], ['substanc
rtling'], ['fact'], ['detailed'], ['fail'], ['impression'], ['fancy'], ['phantasm'], ['come'], ['spirit'], ['seem'],
et'], ['similar'], ['supply'], ['night'], ['stupifi'], ['infamy'], ['black'], ['repose'], ['immense'], ['gin'], ['fu
```

ⓒ 2018. SNU Growth Hackers all rights reserved

Growth Hackers

2. 사전, corpus 생성

```
# Create Dictionary
id2word = corpora.Dictionary(data_lemmatized)
# print(id2word)

# Create Corpus
texts = data_lemmatized
for i in texts:
    try:
        i == [] or ['₩n']
        texts.remove(i)
    except IndexError:
        pass

list_a = []
for i in texts:
    try:
        list_a.append(i[0])
    except IndexError:
        pass

# Term Document Frequency
corpus = [id2word.doc2bow(list_a) for text in texts]
# print(corpus)
# print(corpus[2]) # 일부가 출력되었음

# Human readable format of corpus (term-frequency)
[[(id2word[id], freq) for id, freq in cp] for cp in corpus[:1]]
```

```
[[(1, 1), (3, 1), (5, 1), (7, 1), (9, 1), (11, 1), (13, 1), (15, 1), (17, 1), (19, 1), (21, 1), (23, 1), (25, 1), (27, 1), (29, 1), (31, 1)
2, 1), (44, 1), (46, 2), (48, 1), (50, 1), (52, 1), (53, 1), (55, 1), (57, 1), (59, 1), (60, 1), (62, 1), (64, 1), (66, 1), (69, 1), (71, 1
(84, 1), (85, 1), (86, 1), (88, 1), (90, 1)], [(1, 1), (3, 1), (5, 1), (7, 1), (9, 1), (11, 1), (13, 1), (15, 1), (17, 1), (19, 1), (21, 1)
3, 1), (35, 1), (36, 1), (38, 1), (40, 1), (42, 1), (44, 1), (46, 2), (48, 1), (50, 1), (52, 1), (53, 1), (55, 1), (57, 1), (59, 1), (60, 1
(73, 1), (75, 1), (79, 1), (81, 1), (82, 1), (84, 1), (85, 1), (86, 1), (88, 1), (90, 1)], [(1, 1), (3, 1), (5, 1), (7, 1), (9, 1), (11, 1)
3, 1), (25, 1), (27, 1), (29, 1), (31, 1), (33, 1), (35, 1), (36, 1), (38, 1), (40, 1), (42, 1), (44, 1), (46, 2), (48, 1), (50, 1), (52, 1
(62, 1), (64, 1), (66, 1), (69, 1), (71, 1), (73, 1), (75, 1), (79, 1), (81, 1), (82, 1), (84, 1), (85, 1), (86, 1), (88, 1), (90, 1)], [(1
3, 1), (15, 1), (17, 1), (19, 1), (21, 1), (23, 1), (25, 1), (27, 1), (29, 1), (31, 1), (33, 1), (35, 1), (36, 1), (38, 1), (40, 1), (42, 1
(53, 1), (55, 1), (57, 1), (59, 1), (60, 1), (62, 1), (64, 1), (66, 1), (69, 1), (71, 1), (73, 1), (75, 1), (79, 1), (81, 1), (82, 1), (84,
1), (3, 1), (5, 1), (7, 1), (9, 1), (11, 1), (13, 1), (15, 1), (17, 1), (19, 1), (21, 1), (23, 1), (25, 1), (27, 1), (29, 1), (31, 1), (33,
(44, 1), (46, 2), (48, 1), (50, 1), (52, 1), (53, 1), (55, 1), (57, 1), (59, 1), (60, 1), (62, 1), (64, 1), (66, 1), (69, 1), (71, 1), (73,
(85, 1), (86, 1), (88, 1), (90, 1)], [(1, 1), (3, 1), (5, 1), (7, 1), (9, 1), (11, 1), (13, 1), (15, 1), (17, 1), (19, 1), (21, 1), (23, 1)
5, 1), (36, 1), (38, 1), (40, 1), (42, 1), (44, 1), (46, 2), (48, 1), (50, 1), (52, 1), (53, 1), (55, 1), (57, 1), (59, 1), (60, 1), (62, 1
(75, 1), (79, 1), (81, 1), (82, 1), (84, 1), (85, 1), (86, 1), (88, 1), (90, 1)], [(1, 1), (3, 1), (5, 1), (7, 1), (9, 1), (11, 1), (13, 1)
5, 1), (27, 1), (29, 1), (31, 1), (33, 1), (35, 1), (36, 1), (38, 1), (40, 1), (42, 1), (44, 1), (46, 2), (48, 1), (50, 1), (52, 1), (53, 1
(64, 1), (66, 1), (69, 1), (71, 1), (73, 1), (75, 1), (79, 1), (81, 1), (82, 1), (84, 1), (85, 1), (86, 1), (88, 1), (90, 1)], [(1, 1), (3,
```

Corpus : (word_id, word_frequency)의 매핑

사람이 이해 가능한 형태 →

```
[[('attempt', 1),
  ('growth', 1),
  ('intelligence', 1),
  ('change', 1),
  ('malevolence', 1),
  ('expect', 1),
  ('inclination', 1),
  ('blaze', 1),
  ('collect', 1),
  ('neck', 1),
  ('carcass', 1),
  ('remorse', 1),
  ('top', 1),
  ('white', 1),
  ('great', 1),
  ('odious', 1),
  ('dress', 1),
  ('attention', 1)
```

ⓒ 2018. SNU Growth Hackers all rights reserved

Growth Hackers

3. 모델 생성

```
# Build LDA model
lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
id2word=id2word,
num_topics=20,
random_state=100,
update_every=1,
chunksize=100,
passes=10,
alpha='auto',
per_word_topics=True)
```

© 2018. SNU Growth Hackers all rights reserved

3. 모델생성 - 토픽보기

- 의미?

```
# Print the Keyword in the 10 topics
pprint(lda_model.print_topics())
doc_lda = lda_model[corpus]
```

```
[(0,
  '0.014*"wall" + 0.009*"day" + 0.009*"beast" + 0.008*"house" + '
  '0.008*"plaster" + 0.007*"frequent" + 0.007*"length" + 0.006*"go" + '
  '0.006*"animal" + 0.006*"pluto"'),
 (1,
  '0.013*"house" + 0.012*"wall" + 0.009*"day" + 0.008*"night" + 0.008*"animal" '
  '+ 0.007*"length" + 0.007*"beast" + 0.007*"go" + 0.007*"leave" + '
  '0.006*"destroy"'),
```

토픽0에 기여하는 상위 10개의 키워드를 나타낸다. Wall의 가중치는 0.014, day의 가중치는 0.009

ⓒ 2018. SNU Growth Hackers all rights reserved

4. 모델 평가

```
# Compute Perplexity(모델 난이도)
print('₩nPerplexity: ', lda_model.log_perplexity(corpus)) # a measure of how good the model is. lower the better.

# Compute Coherence Score(일관성 점수)
coherence_model_lda = CoherenceModel(model=lda_model, texts=data_lemmatized, dictionary=id2word, coherence='c_v')

coherence_lda = coherence_model_lda.get_coherence()
print('₩nCoherence Score: ', coherence_lda)
```

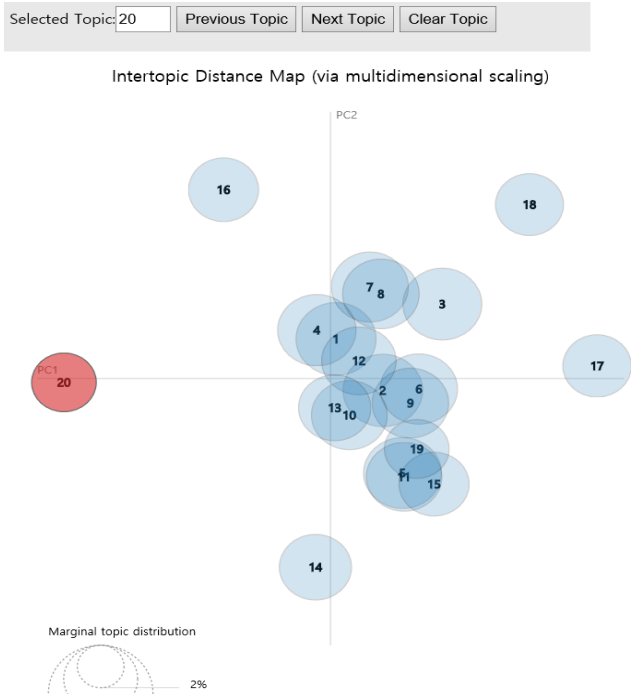
```
Perplexity:  -6.177489118961218

Coherence Score:  0.8646799430899037
```

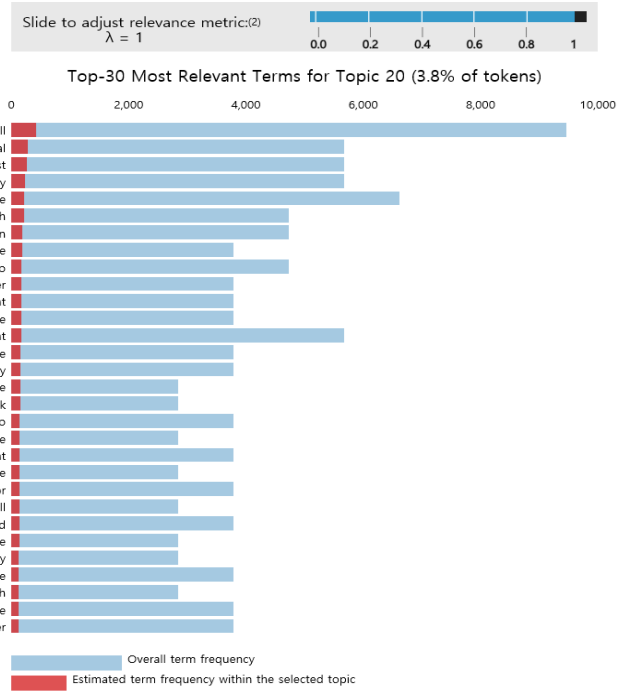
ⓒ 2018. SNU Growth Hackers all rights reserved

4. 모델 평가 - 모델 시각화

```
# Visualize the topics(대화형차트) - 토픽과 키워드 검사
pyLDAvis.enable_notebook()
vis = pyLDAvis.gensim.prepare(lda_model, corpus, id2word)
vis
```

- 버블이 클 수록 그 토픽이 더 일반적.
- 좋은 토픽 모델은 하나의 사분면에 클러스터되지 않고 차트 전체에 흩어져있는 상당히 크고 겹치지 않는 버블을 가질 것.
- 너무 많은 토픽을 가진 모델은 일반적으로 차트의 한 영역에 클러스터된 작은 크기의 버블이 많이 겹치게 될 것





© 2018. SNU Growth Hackers all rights reserved

Growth Hackers

4. 모델 평가

```
# Show Topics
pprint(lda_model.show_topics(formatted=False))
```

```
[(16,
  [('wall', 0.015863571),
   ('animal', 0.0103756795),
   ('beast', 0.009993294),
   ('day', 0.008844533),
   ('house', 0.008187886),
   ('length', 0.008043507),
   ('reason', 0.0071598534),
   ('come', 0.0070280326),
   ('go', 0.006521989),
   ('character', 0.0064299614)]),
 (0,
  [('wall', 0.014334574),
   ('day', 0.009163644),
   ('beast', 0.00870448),
   ('house', 0.008245798),
   ('plaster', 0.00800458),
   ('frequent', 0.007403196),
```

ⓒ 2018. SNU Growth Hackers all rights reserved

Growth Hackers

5. k 수 찾기

```
# k의 수(최적의 토픽 수) 를 찾아보기
#밑의 함수는 여러 LDA 모델을 교육하고 모델 및 해당하는 일관성 점수를 알려준다.
def compute_coherence_values(dictionary, corpus, texts, limit, start=2, step=3):
    """
    Compute c_v coherence for various number of topics

    Parameters:
    ----------
    dictionary : Gensim dictionary
    corpus : Gensim corpus
    texts : List of input texts
    limit : Max num of topics

    Returns:
    -------
    model_list : List of LDA topic models
    coherence_values : Coherence values corresponding to the LDA model with respective number of topics
    """
    coherence_values = []
    model_list = []
    for num_topics in range(start, limit, step):
        model=gensim.models.ldamodel.LdaModel(corpus=corpus,id2word=id2word,num_topics=num_topics,random_state=100,
                                              update_every=1,chunksize=100,passes=10,alpha='auto',per_word_topics=True)
        model_list.append(model)
        coherencemodel = CoherenceModel(model=model, texts=texts, dictionary=dictionary, coherence='c_v')
        coherence_values.append(coherencemodel.get_coherence())

    return model_list, coherence_values
```

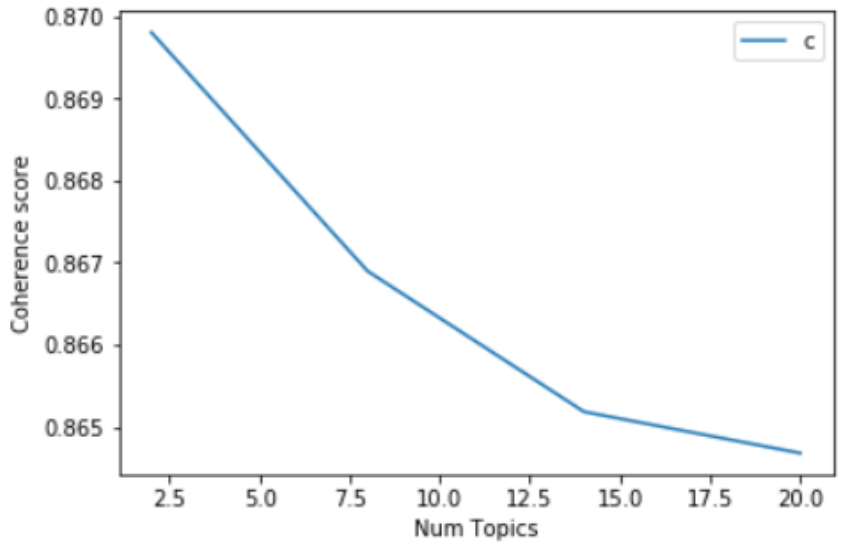
```
# Can take a long time to run.
model_list, coherence_values = compute_coherence_values(dictionary=id2word, corpus=corpus, texts=data_lemmatized, start=2, limit=40, step=6)
```

© 2018. SNU Growth Hackers all rights reserved

Growth Hackers

5. K 수 찾기

```
# Show graph
limit=40; start=2; step=6;
x = range(start, limit, step)
plt.plot(x, coherence_values)
plt.xlabel("Num Topics")
plt.ylabel("Coherence score")
plt.legend(("coherence_values"), loc='best')
plt.show()
```




```
# Print the coherence scores 가장 일관성 높은 모델 선택
for m, cv in zip(x, coherence_values):
    print("Num Topics =", m, " has Coherence Value of", round(cv, 4))
```

```
Num Topics = 2  has Coherence Value of 0.8698
Num Topics = 8  has Coherence Value of 0.8669
Num Topics = 14  has Coherence Value of 0.8652
Num Topics = 20  has Coherence Value of 0.8647
Num Topics = 26  has Coherence Value of nan
Num Topics = 32  has Coherence Value of nan
Num Topics = 38  has Coherence Value of nan
```

ⓒ 2018. SNU Growth Hackers all rights reserved

Growth Hackers

5. K 수 찾기

```
# Select the model and print the topics
optimal_model = model_list[0] #여긴 경우에 따라 고치기
model_topics = optimal_model.show_topics(formatted=False)
pprint(optimal_model.print_topics(num_words=10))
```

```
[(0,
  '0.014*"wall" + 0.009*"beast" + 0.008*"day" + 0.007*"animal" + 0.007*"house" '
  '+ 0.007*"night" + 0.007*"minute" + 0.007*"reason" + 0.006*"plaster" + '
  '0.006*"length"'),
 (1,
  '0.013*"wall" + 0.012*"house" + 0.009*"night" + 0.009*"animal" + 0.008*"day" '
  '+ 0.008*"beast" + 0.007*"go" + 0.007*"length" + 0.007*"reason" + '
  '0.006*"cat"')]
```

ⓒ 2018. SNU Growth Hackers all rights reserved