자연어 처리를 활용한 한국어 유튜브 댓글 종류 분류

목차

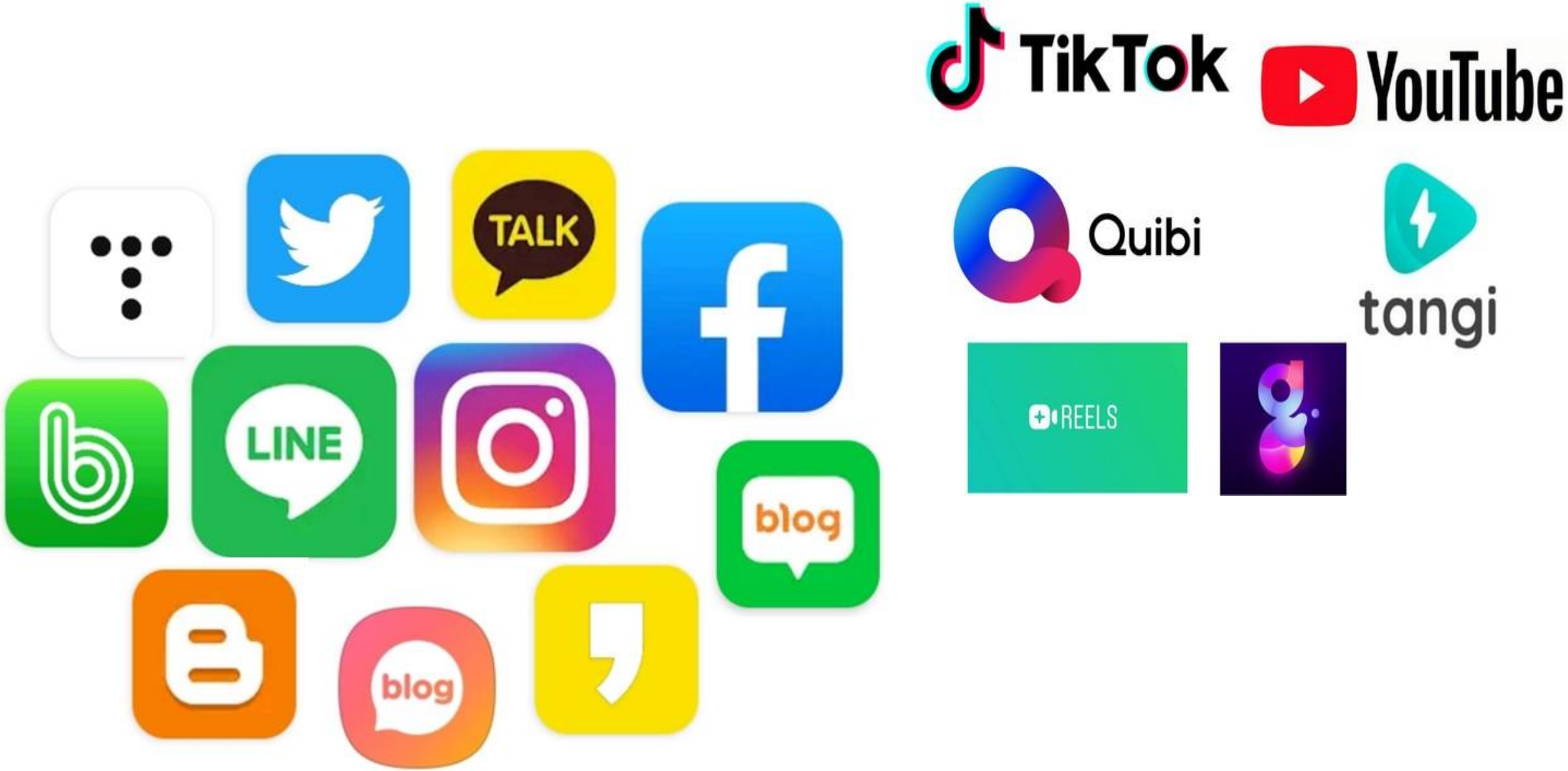
Part 1

프로젝트 소개

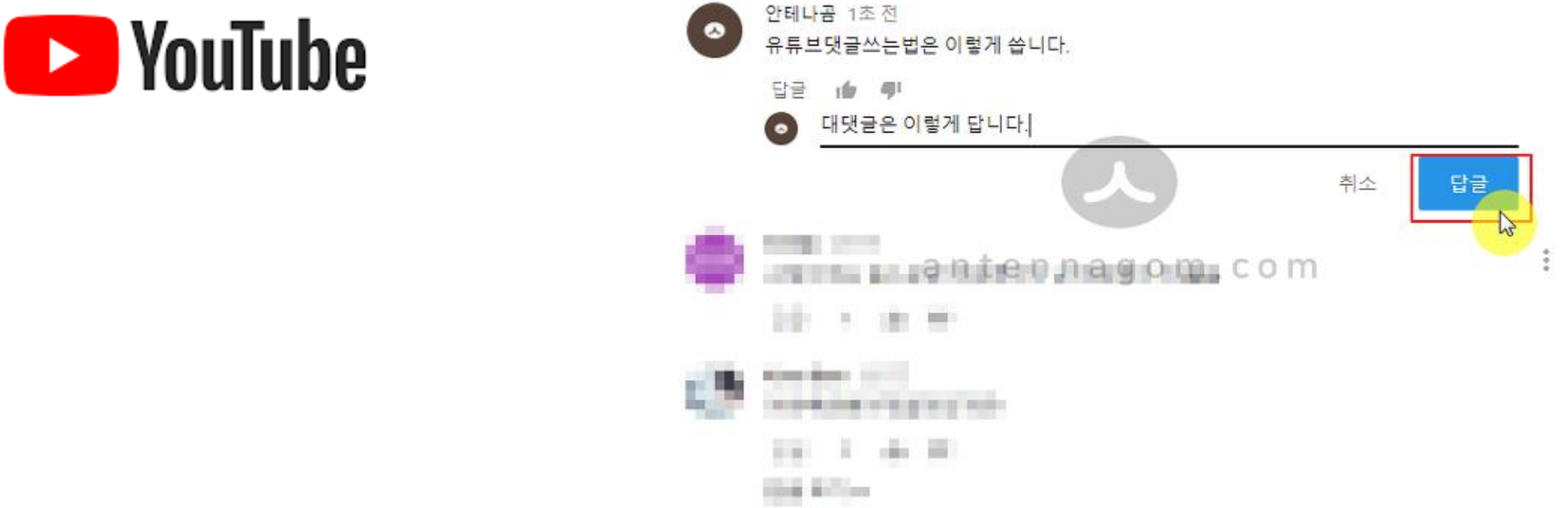
1.1 개요

1.2 기능

1.3 구조


TikTok
YouTube
Quibi
tangi
REELS
TALK
LINE
blog
blog

Part 1.1 개요



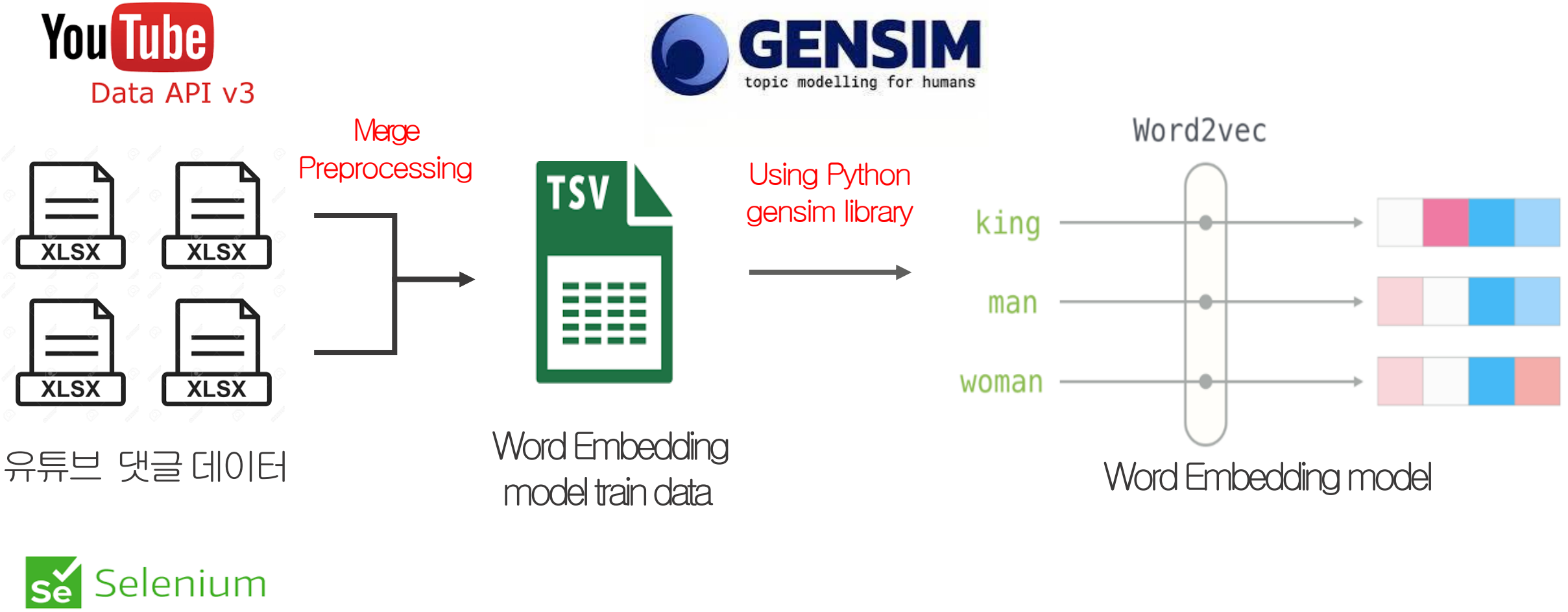
Part 1.2 기능

분류 데이터 수집
자연어 전처리

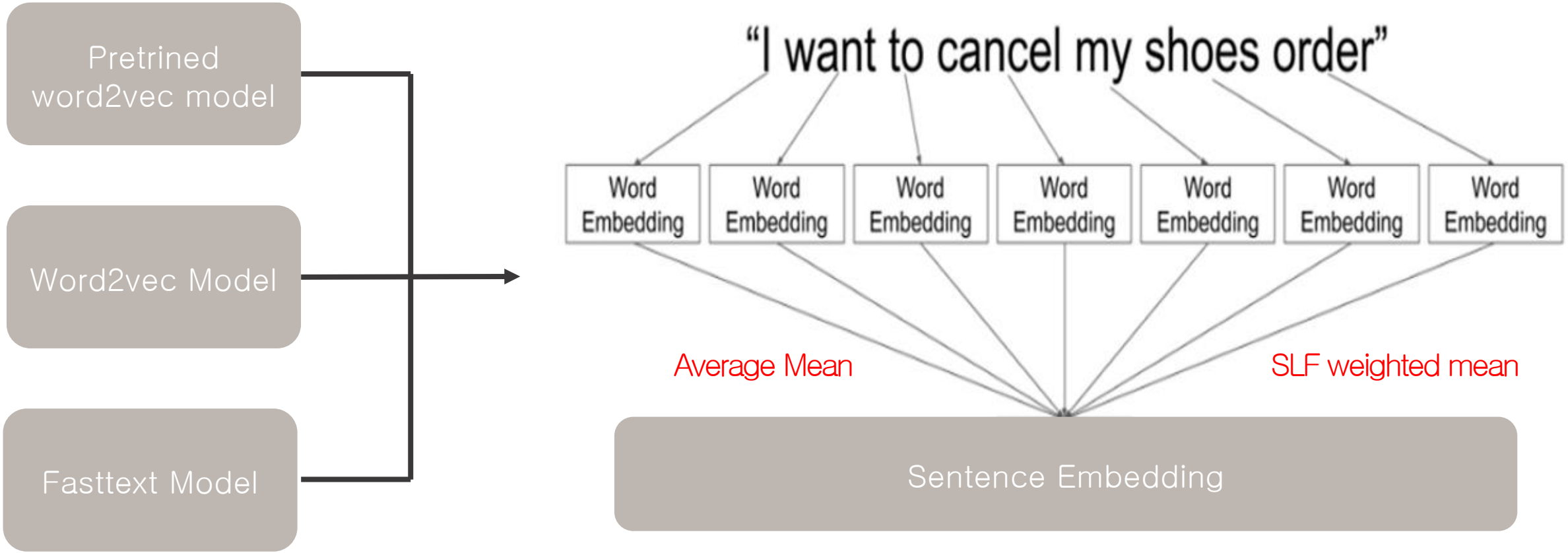
Word Embedding model을 기반으로 문장을 연속적인 vector값으로 표현하는 모델을 구현

구현한 모델을 통해
분류 데이터 클러스터링,
차원 축소와 시각화

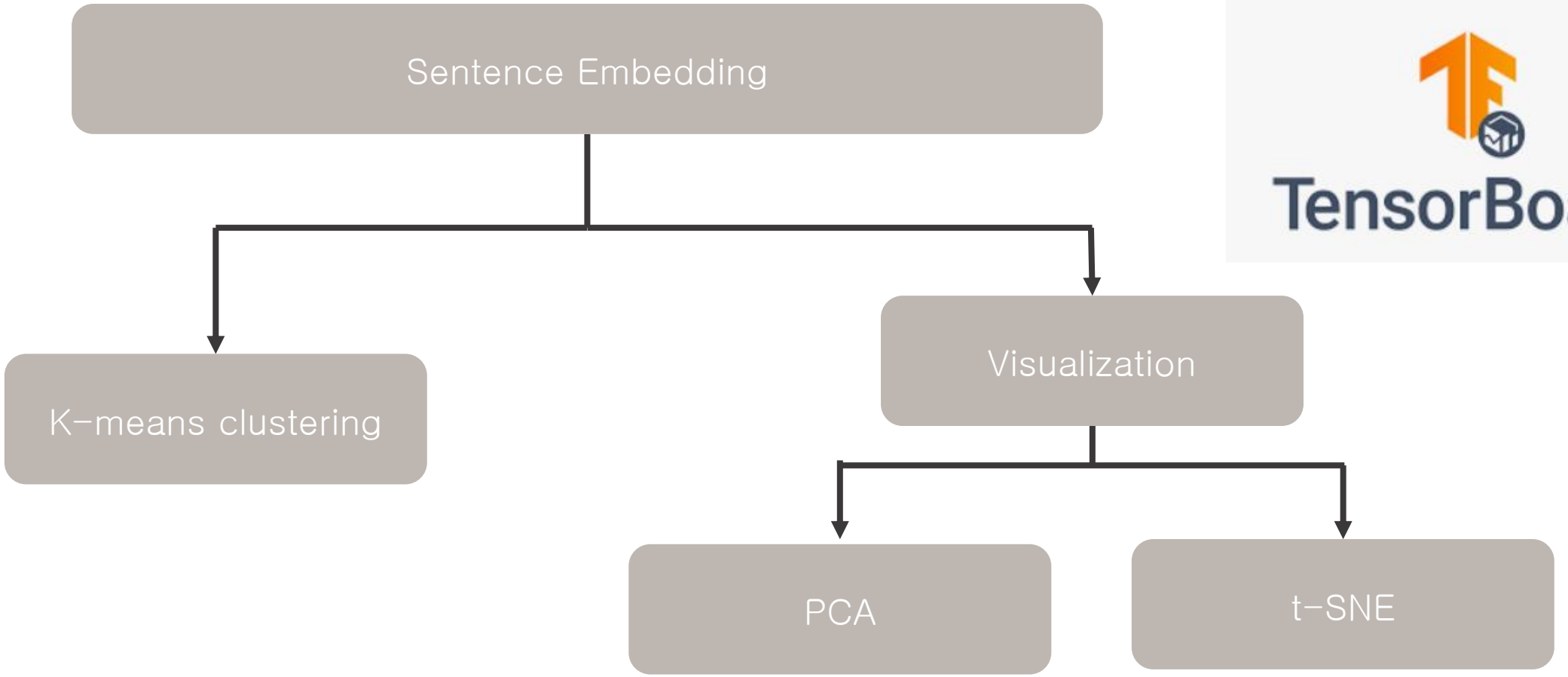
Part 1.3 구조 : Data scraping & Create Word Embedding model



Part 1.3 구조 : Sentence to Embedding vector model



Part 1.3 구조 : Clustering & Visualization(PCA, t-SNE)



TensorBoard

Part 2

진행 상황 · 최종 결과

2.1 Data scraping & Preprocessing

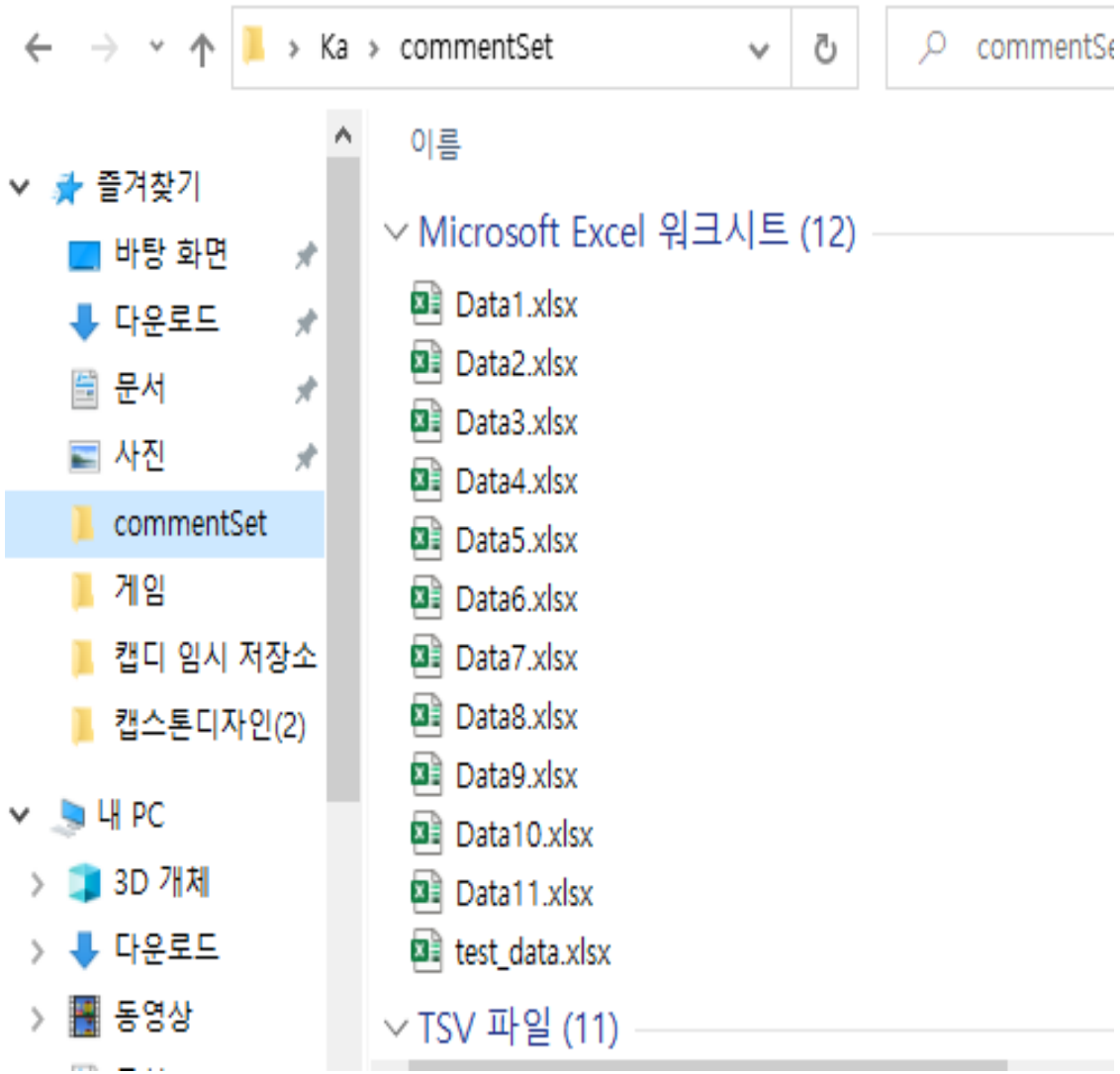
2.2 Word & Sentence Embedding model

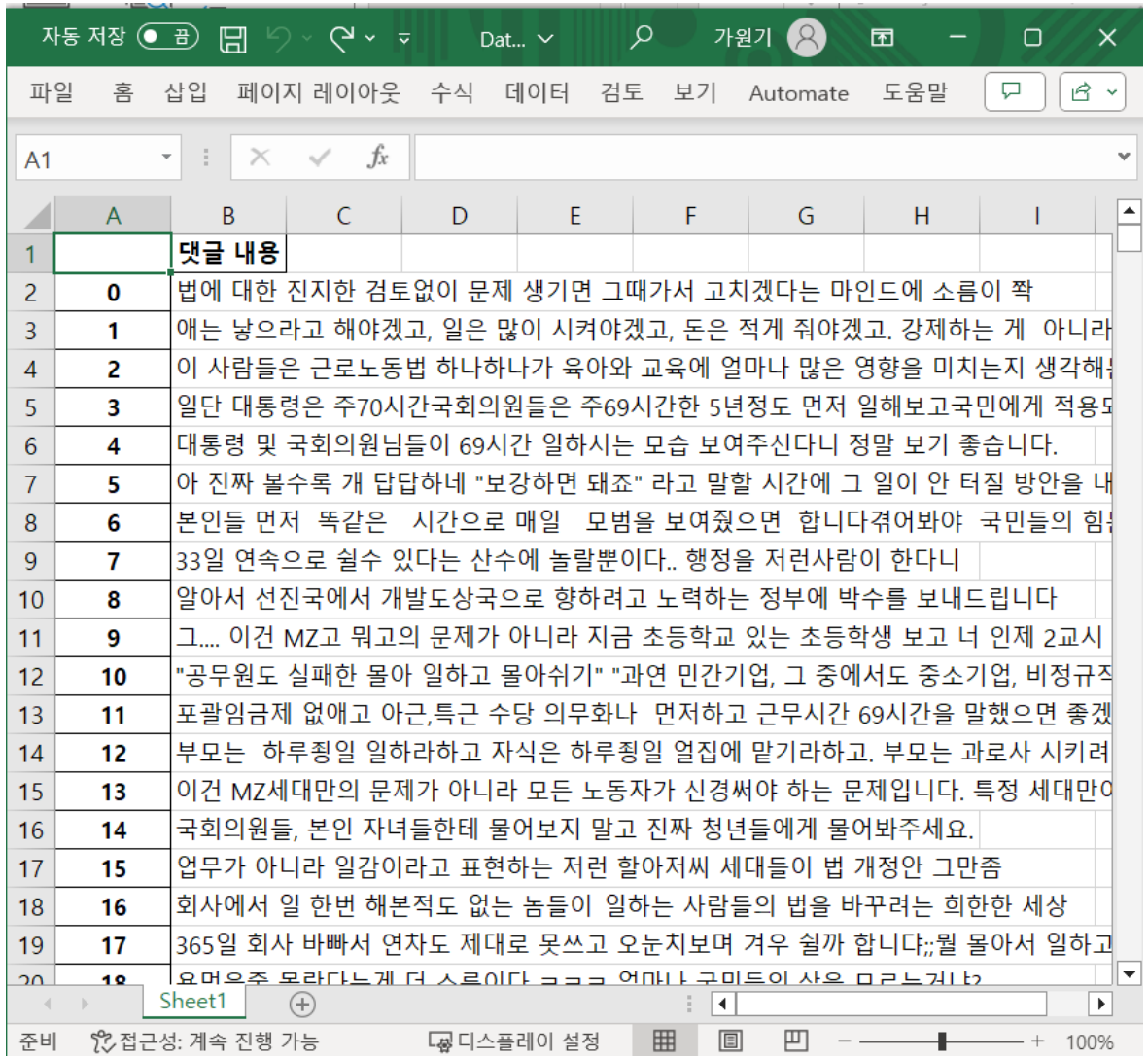
2.3 Clustering & Visualization

video id를 입력하면 모든 댓글 수집가능
사용 가능 할당량이 제한되어 있다.

URL을 입력하면 브라우저를 자동으로
동작하여 댓글을 로딩하고 수집한다.

Part 2.1 Data scraping





	A	댓글 내용
2	0	법에 대한 진지한 검토없이 문제 생기면 그때가서 고치겠다는 마인드에 소름이 쫙
3	1	애는 낳으라고 해야겠고, 일은 많이 시켜야겠고, 돈은 적게 줘야겠고. 강제하는 게 아니라
4	2	이 사람들은 근로노동법 하나하나가 육아와 교육에 얼마나 많은 영향을 미치는지 생각해
5	3	일단 대통령은 주70시간국회의원들은 주69시간한 5년정도 먼저 일해보고국민에게 적용도
6	4	대통령 및 국회의원님들이 69시간 일하시는 모습 보여주신다니 정말 보기 좋습니다.
7	5	아 진짜 볼수록 개 답답하네 "보강하면 돼죠" 라고 말할 시간에 그 일이 안 터질 방안을 내
8	6	본인들 먼저 똑같은 시간으로 매일 모범을 보여줬으면 합니다겨어봐야 국민들의 힘
9	7	33일 연속으로 쉴수 있다는 산수에 놀랄뿐이다.. 행정을 저런사람이 한다니
10	8	알아서 선진국에서 개발도상국으로 향하려고 노력하는 정부에 박수를 보내드립니다
11	9	그.... 이건 MZ고 뭐고의 문제가 아니라 지금 초등학교 있는 초등학생 보고 너 인제 2교시
12	10	"공무원도 실패한 몰아 일하고 몰아쉬기" "과연 민간기업, 그 중에서도 중소기업, 비정규직
13	11	포괄임금제 없애고 아근,특근 수당 의무화나 먼저하고 근무시간 69시간을 말했으면 좋겠
14	12	부모는 하루죙일 일하라하고 자식은 하루죙일 얼집에 맡기라하고. 부모는 과로사 시키려
15	13	이건 MZ세대만의 문제가 아니라 모든 노동자가 신경써야 하는 문제입니다. 특정 세대만이
16	14	국회의원들, 본인 자녀들한테 물어보지 말고 진짜 청년들에게 물어봐주세요.
17	15	업무가 아니라 일감이라고 표현하는 저런 할아저씨 세대들이 법 개정안 그만좀
18	16	회사에서 일 한번 해본적도 없는 놈들이 일하는 사람들의 법을 바꾸려는 희한한 세상
19	17	365일 회사 바빠서 연차도 제대로 못쓰고 오눈치보며 겨우 쉴까 합니댜;;뭘 몰아서 일하고

Part 2.1 Preprocessing

```
def preprocessing(fileName):
    file_name = fileName
    df = pd.read_excel(file_name)
    data = df["댓글 내용"][:]

    preprocessing_data = []
    pattern = re.compile('[가-힣]+')
    # 한글만 남기고 제거

    for word in data:
        result = pattern.findall(str(word))
        preprocessing_data.append(result)

    #형태소 분석 및 불용어 처리
    okt = Okt()
    stopwords = ['의','가','이','은','들','는','좀','잘','걍','과','도','를','으로',
                 '자','에','와','한','하다', '거', '식', '더', '것']
    tokenized_data = []

    for text in preprocessing_data:
            temp_X = okt.nouns(str(text))
            temp_X = [word for word in temp_X if not word in stopwords]
            temp_X = set(temp_X)
            tokenized_data.append(list(temp_X))

    # 파일로 저장하는건 따로 분리
    # rawtest = np.array(data)
    # rawtest = pd.DataFrame(rawtest)
    # rawtest.to_csv(tsvName, sep='Wt')

    return tokenized_data
```

1. 한글을 제외한 나머지를 제거

2. 한국어 형태소 분석 라이브러리를 활용하여 명사형태로 전환

3. 불용어사전에 저장된 학습에 중요하지 않은 단어를 제거

Part 2.1 Preprocessing

train_data20.tsv - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

	0
0	['불만', '무', '국민', '모습', '솔선수범', '업무', '연중', '주', '시간', '휴가', '은
1	['원래', '사실', '말', '여러분', '거', '노동시간', '가당', '주', '시간', '모자', '지
2	['조직', '책임자', '이번', '책임', '추진', '반성', '고민', '고용노동부', '기회',
3	['문제', '일', '시간', '사람', '정']
4	['답', '일', '불', '지금', '애', '사람', '아이', '진짜']
5	['결정', '수당', '나라', '가족', '이제', '거', '퇴직금', '주', '사람', '한국', '휴가
6	['키', '정말', '아무', '제대로', '주', '시간', '사람', '바짝', '대통령']
7	['주', '파면', '시간', '대통령실', '정말', '시험', '운영', '일단', '정책', '공무원
8	['타', '국가', '주', '한국', '달', '부분', '노동시간', '매번', '왜', '그냥', '더', '저
9	['것', '때', '정말', '평균', '그것', '계속', '노동시간', '헛', '저', '일', '희망', '싦
10	['정치가', '계속', '말', '경험', '공장', '최저', '일', '찬성', '만약', '달', '살']
11	['시간', '번', '제나', '개선', '머리', '안이', '주', '포괄', '임금', '진짜', '사람']
12	['국민', '힘', '의원', '일', '시간']
13	['벌써', '퇴사', '적', '출근', '프레', '경험', '후유증', '스', '다른', '처음', '당시
14	['나올떄', '신분', '솔선수범', '먼저', '법안', '기획', '시행', '추진', '달', '진짜

Ln 1, Col 1 100% Windows (CRLF) UTF-8

```
In [39]: counter
```

```
Out[39]: Counter({'불만': 624,
                  '무': 116,
                  '국민': 2284,
                  '모습': 255,
                  '솔선수범': 52,
                  '업무': 96,
                  '연중': 2,
                  '주': 692,
                  '시간': 1848,
                  '휴가': 177,
                  '임기': 40,
                  '대통령': 658,
                  '동안': 165,
                  '원래': 122,
                  '사실': 275,
                  '말': 1947,
                  '여러분': 90,
                  '거': 1915,
                  '노동시간': 88,
```

학습에 사용한 데이터셋에서
단어가 얼마나 자주 등장하는지 확인
조사등 의미없는 단어가 남아있다면
불용어 사전에 포함시켜서 제외

Part 2.2 Word Embedding model

```
In [7]: kovec.wv.most_similar(positive=['일본', '서울'], negative=['한국'])
```

```
Out[7]: [('도쿄', 0.4962024092674255),
         ('영등포', 0.4607112407684326),
         ('서울특별시', 0.45662832260131836),
         ('경성', 0.44781729578971863),
         ('아현동', 0.4475313723087311),
         ('경성부', 0.4472092390060425),
         ('세종로', 0.44181060791015625),
         ('혜화동', 0.44022461771965027),
         ('원효로', 0.4394114017486572),
         ('상도동', 0.4373798370361328)]
```

```
In [173]: kovec.wv.vocab
```

```
Out[173]: {'관위': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd037754f0>,
           '정어리': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd037754c0>,
           '유식론': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd03775580>,
           '장로회': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd037755e0>,
           '춘추관': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd03775640>,
           '도입부': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd037756a0>,
           '민병': <gensim.models.deprecated.keyedvectors.Vocab at 0x1bd03775700>,
```

pre-trained model 사용

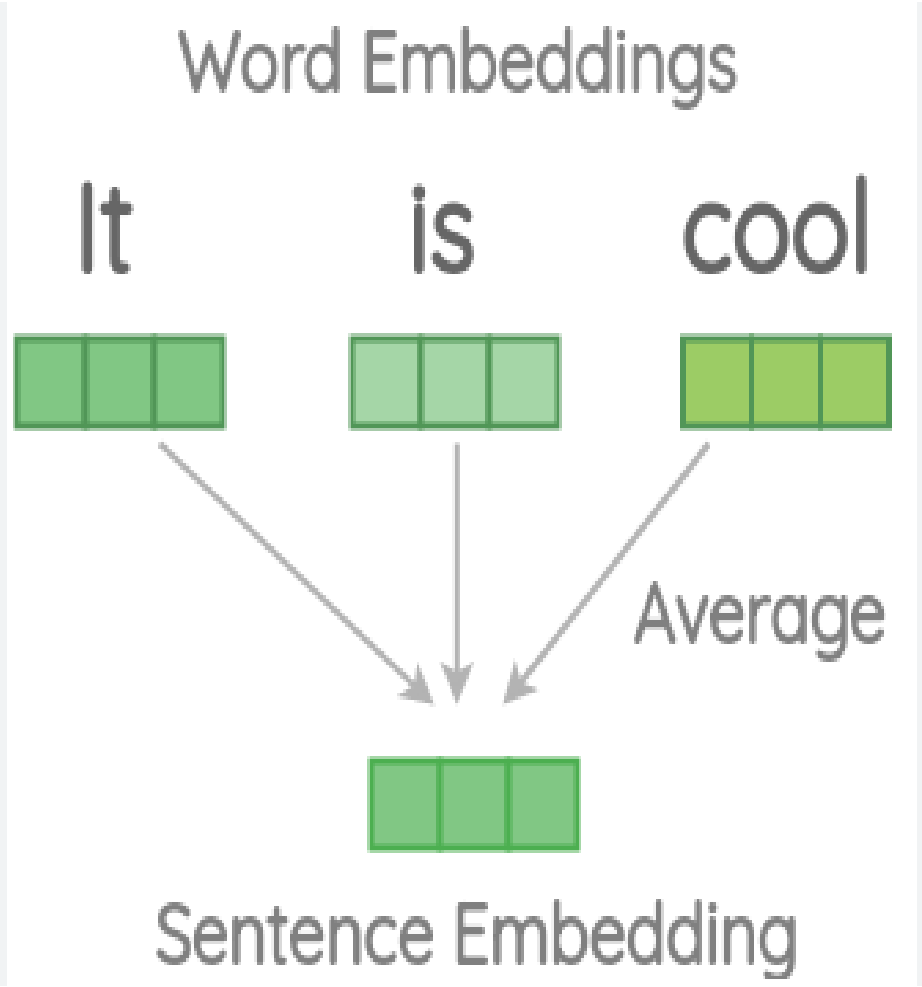
```
In [19]: w2vmodel20= Word2Vec.load('wordModels###w2vmodel20')
         w2vmodel20.wv.most_similar('오염수')
```

```
Out[19]: [('한국', 0.9277608394622803),
          ('미래', 0.9033635854721069),
          ('핵', 0.9004865884780884),
          ('아주', 0.8915683031082153),
          ('대표', 0.8894720077514648),
          ('끼리', 0.8801928758621216),
          ('친일', 0.8782353401184082),
          ('공업', 0.8780603408813477),
          ('국가', 0.8778339624404907),
          ('오히려', 0.8752624988555908)]
```

```
In [20]: w2vmodel20.wv.vocab
```

```
Out[20]: {'불만': <gensim.models.keyedvectors.Vocab at 0x19d08d87970>,
          '무': <gensim.models.keyedvectors.Vocab at 0x19d08d87a00>,
          '국민': <gensim.models.keyedvectors.Vocab at 0x19d08d87b50>,
          '모습': <gensim.models.keyedvectors.Vocab at 0x19d0867ceb0>,
          '솔선수범': <gensim.models.keyedvectors.Vocab at 0x19d0867ce20>,
          '업무': <gensim.models.keyedvectors.Vocab at 0x19d0867ccd0>,
          '연중': <gensim.models.keyedvectors.Vocab at 0x19d0867cd30>,
```

댓글 데이터로 학습시킨 model사용




```python
def sentenceEmbedding(kovec,tokenized_data):
    # 0 vector를 만들기
    sentence_vec = (kovec.wv["왜"]*0)

    sentence2vec = []

    for sentence in tokenized_data:
        sentence_vec = (kovec.wv["왜"]*0)
        count=1
        for word in sentence:
            try:
                sentence_vec+=kovec.wv[word]
                count+=1
            except KeyError:
                pass
        #가중치를 사용해서 문장 임베딩하는 방법을 고려중.
        #sentence2vec.append(sentence_vec) 나누지 않는게 더 좋을수도?
        sentence2vec.append(sentence_vec/count)


    return sentence2vec
```

Test Data
(110)

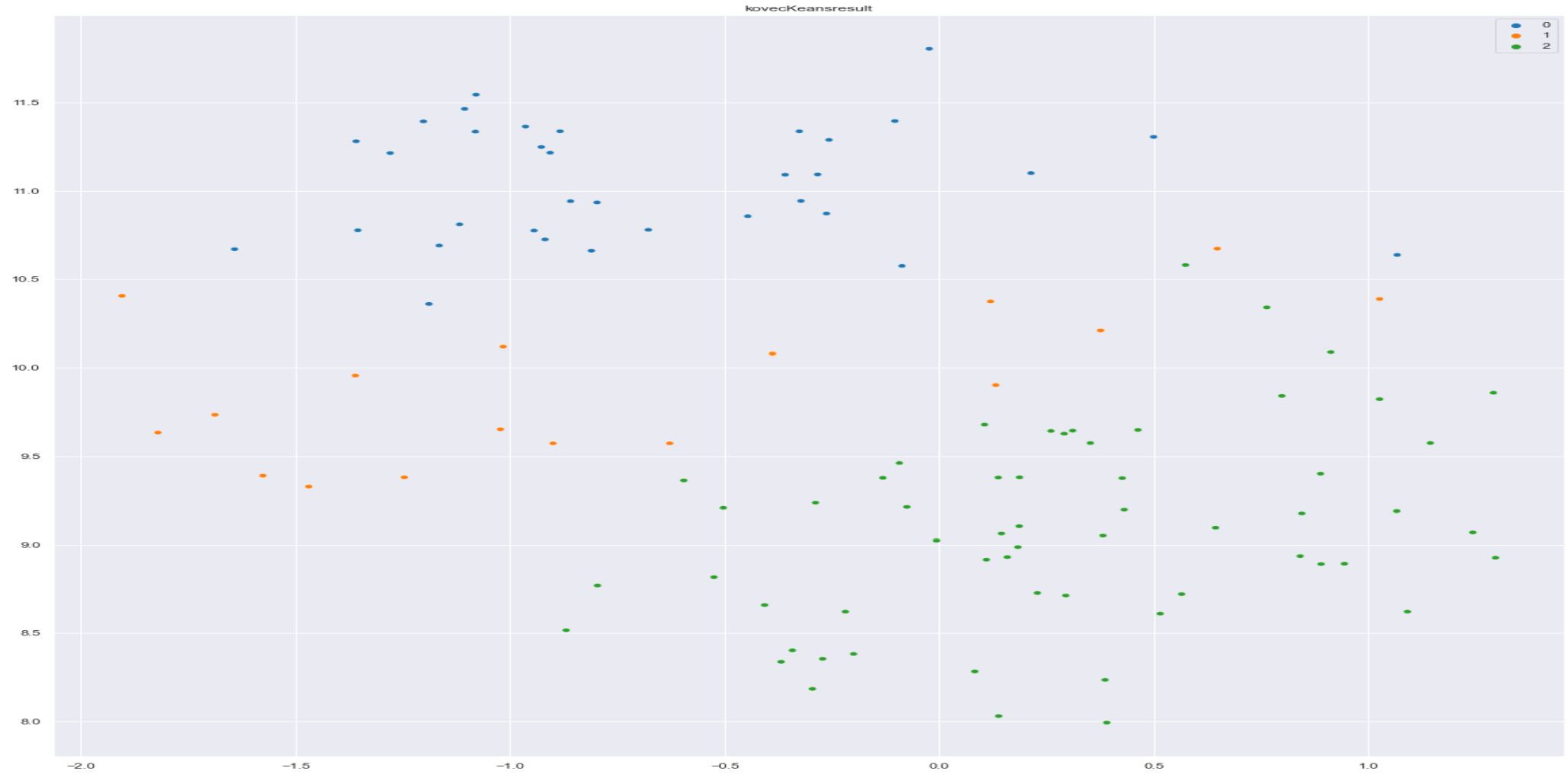
POSITIVE(50)

NEGATIVE(30)

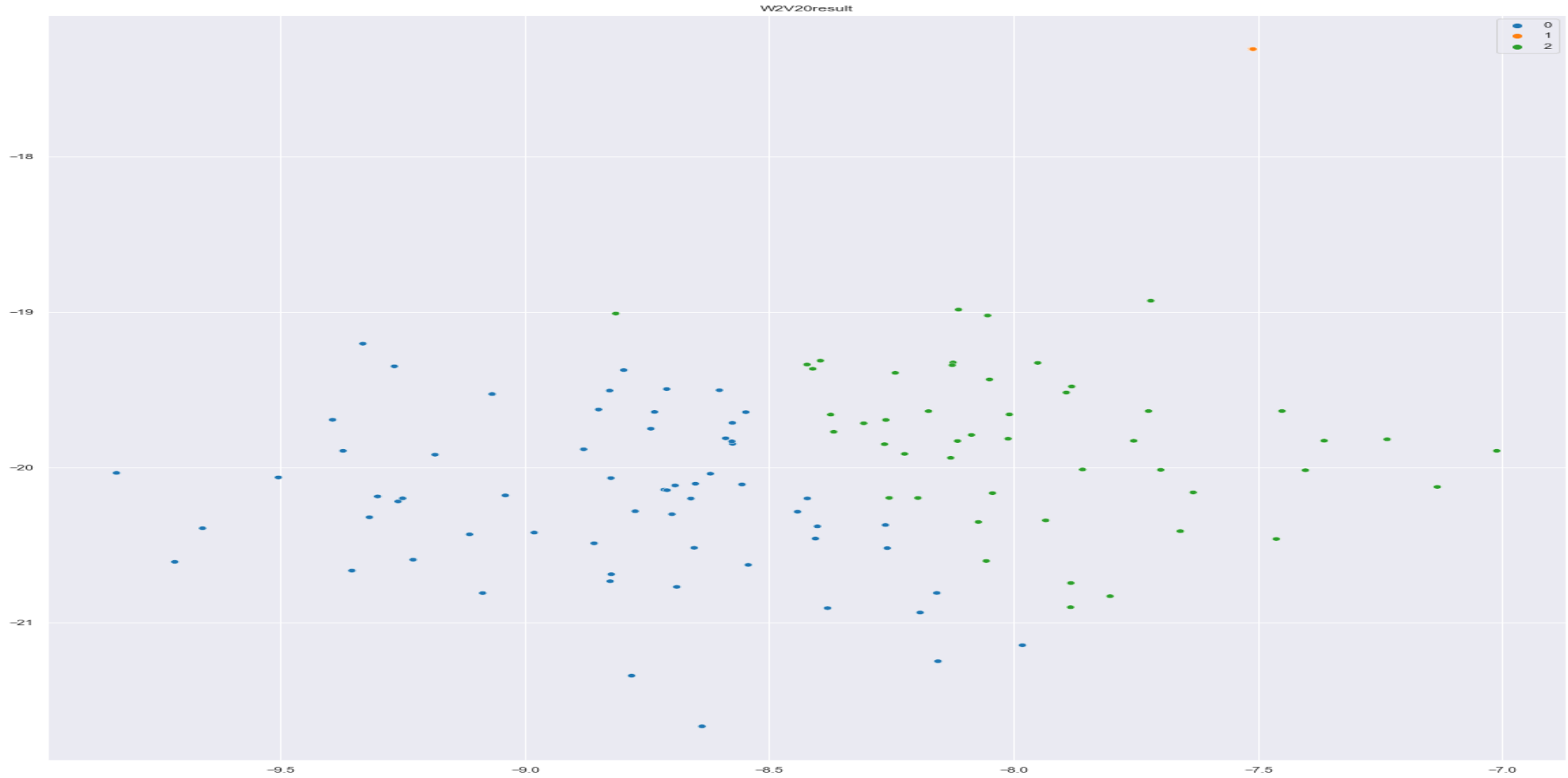
SPAM(30)

```
tokenized_data = preprocessing("commentSet₩₩test_data.xlsx")
sentence2vec = sentenceEmbedding(W2Vmodel, tokenized_data)

k = 3
kmeans = KMeans(n_clusters=k, random_state=2021)
y_pred = kmeans.fit_predict(sentence2vec)

# tsne
tsne = TSNE(verbose=1, perplexity=100, random_state=2021)
X_embedded = tsne.fit_transform(sentence2vec)
print('Embedding shape 확인', X_embedded.shape)

# 시각화
sns.set(rc={'figure.figsize':(20,20)})

sns.set_palette('bright')

sns.scatterplot(X_embedded[:,0], X_embedded[:,1], hue=y_pred,
                legend='full',palette='tab10')     # kmeans로 예측

plt.title('kovecKeansresult')
plt.savefig("kovecKeansresult.png")
plt.show()
```

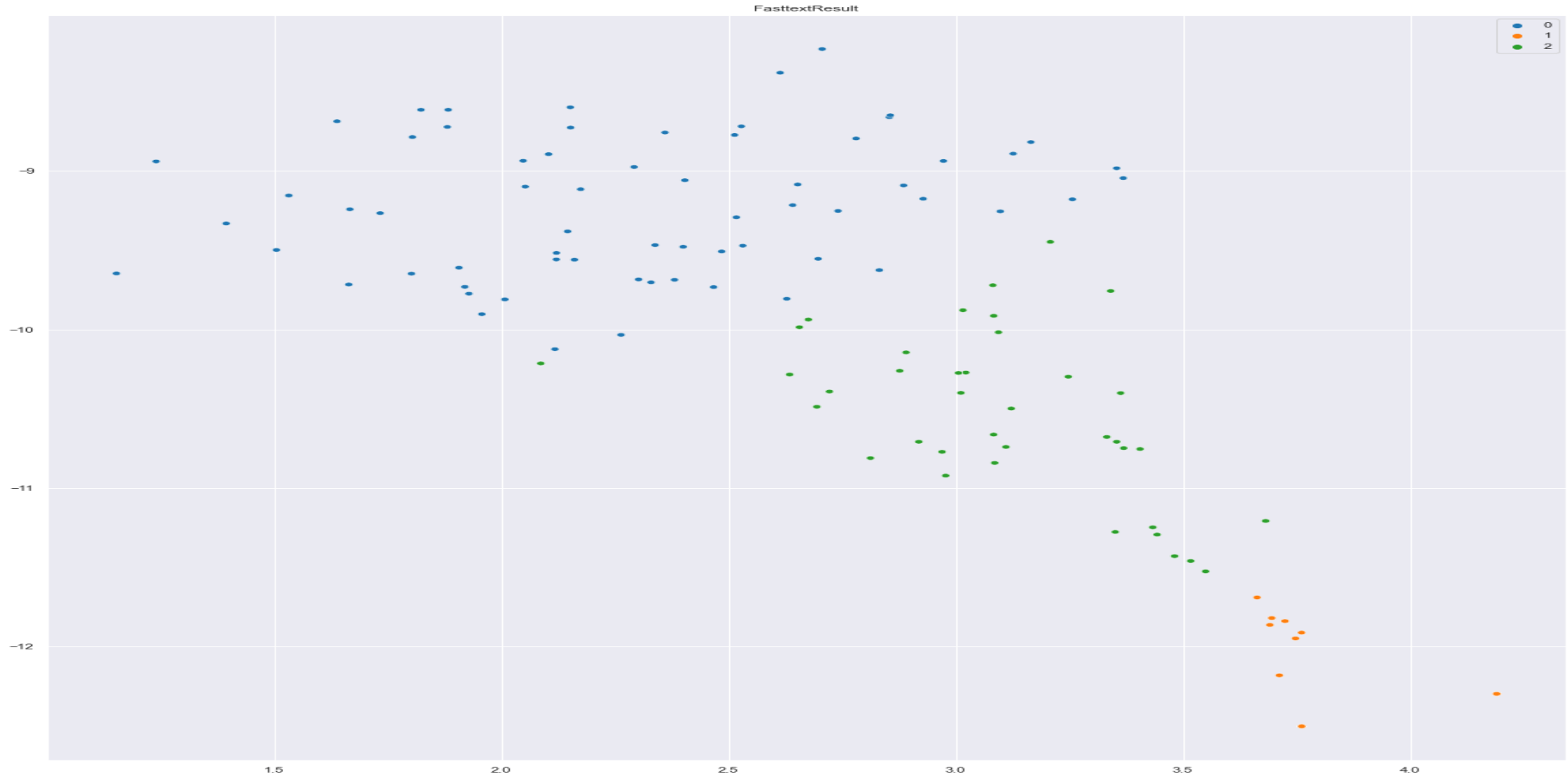
Part 2.3 Clustering (pre-trained Word2Vec model)



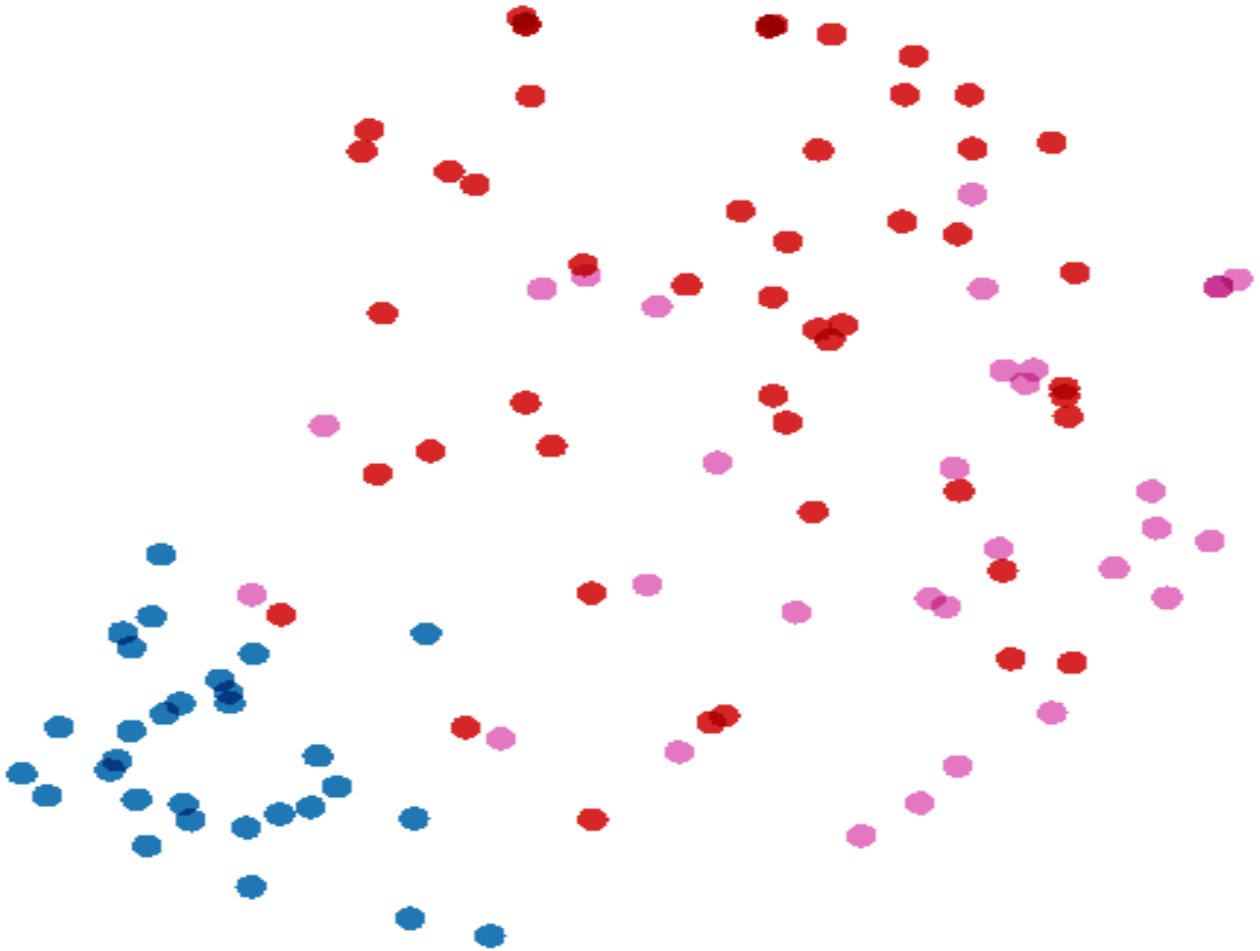
Part 2.3 Clustering (Word2Vec model)



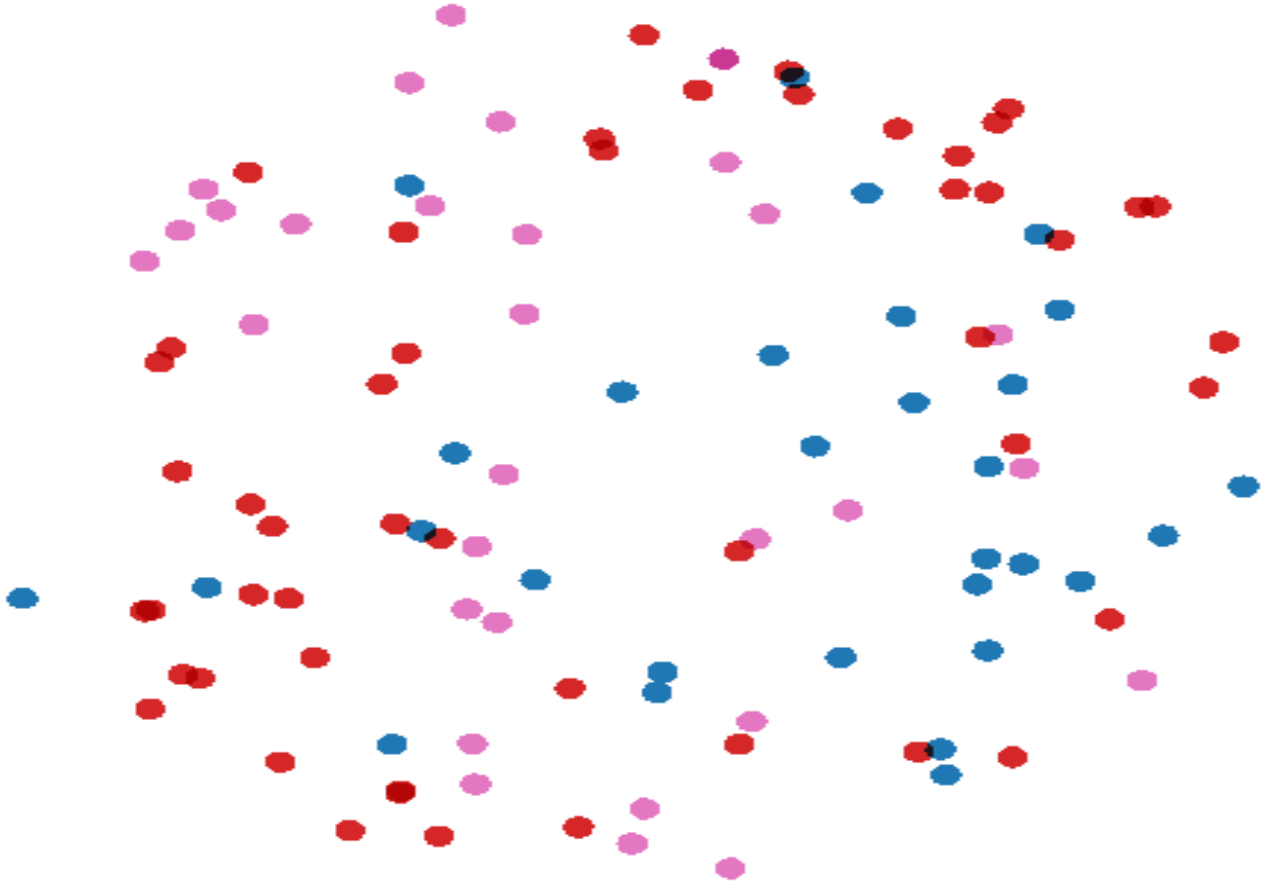
Part 2.3 Clustering (Fasttext model)



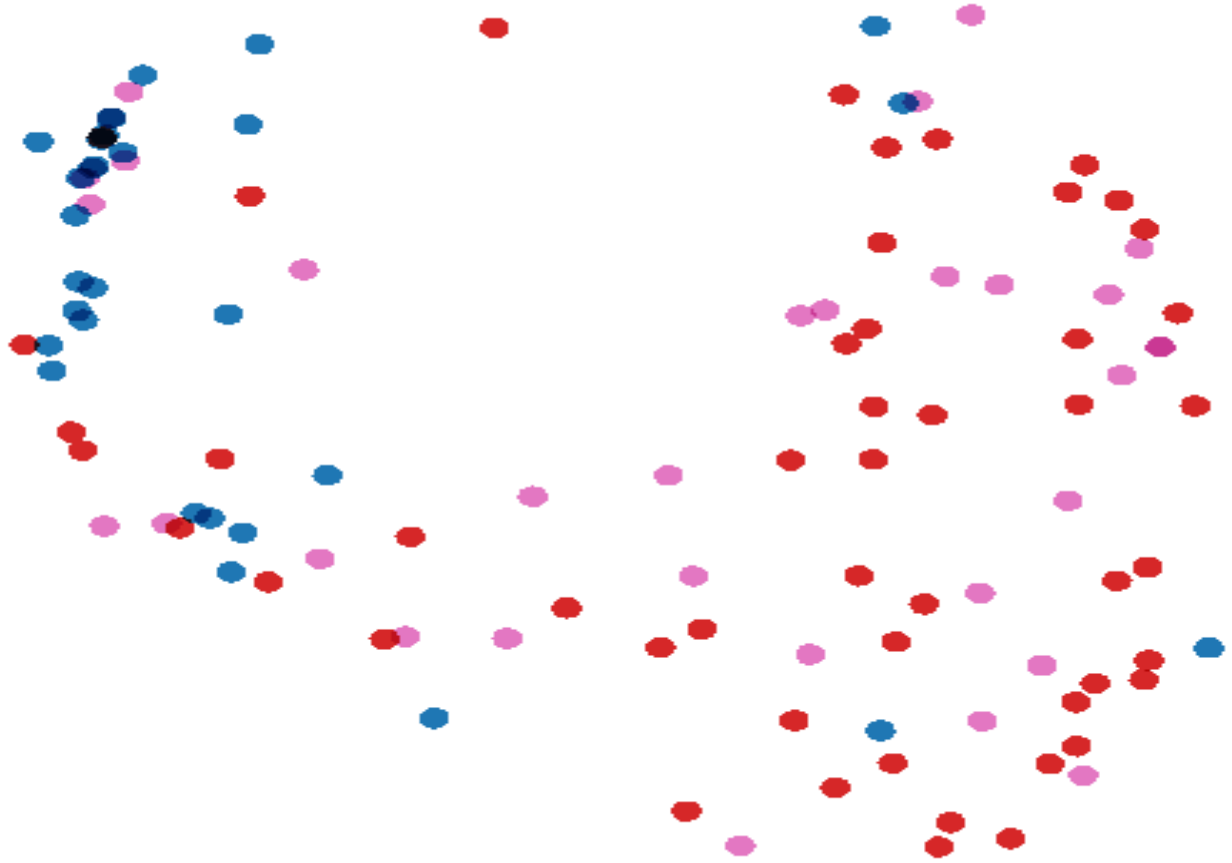
Part 2.3 dimensionality reduction Visualization (pre-trained Word2Vec model)



Part 2.3 dimensionality reduction Visualization (Word2Vec model)



Part 2.3 dimensionality reduction Visualization (Fasttext model)



Part 3

아쉬웠던 점

1. 초기 계획보다 부족한 결과

한 영상에서 수집한 rawdata를 분류하는 데 있어서는 감성분석은 힘들고 댓글 중 스팸 댓글을 구분할 정도의 결과를 보였다.

2. 댓글 데이터로 모델 학습이 계획처럼 잘 이루어지지 않았음

댓글 데이터가 생각했던 것 보다 학습할 때 제외 시켜야 하는 예외 데이터들이 많아서 필요 데이터 수가 더 늘어났다고 생각함.