주제분석 2주차 패키지

- 분석 툴은 R/Python 둘 다 가능합니다. 주제분석 1주차 패키지 문제의 조건 및 힌트는 Python을 기준으로 하지만, R을 사용해도 무방합니다.
- 제출형식은 HTML, PDF 모두 가능합니다. **.ipynb 이나 .R 등의 소스코드 파일은 불가능합니다.** 파일은 psat2009@naver.com으로 보내주세요.
- 패키지 과제 발표는 세미나 쉬는 시간 후에 하게 되며, 랜덤으로 4시 00분에 발표됩니다.
- 제출기한은 **5/4 자정까지** 이고, 지각 시 벌금 5000원, 미제출시 10000원입니다. 패키지 무단 미제출 2회 시 퇴출이니 유의해주세요.

Chapter 1 토큰화

우리가 일상생활에서 사용하는 언어의 의미를 분석하여 컴퓨터가 처리할 수 있도록 하기 위해 많은 과정이 필요합니다. 자연어 처리는 딥러닝에 대한 이해뿐만 아니라 언어에 대한 이해도 필요하며, 언어 종류마다 그 형태가 다양하기 때문에 처리가 어렵습니다. 오늘은 전처리부터 매우 간단한 모델링까지 해볼 것이며 딥러닝팀 클린업 3주차 내용과 흐름이 유사하기 때문에 개념적 이해가 추가적으로 필요하신 분은 참고하시면 됩니다.

먼저 전처리 기본이 되는 토큰화부터 진행해보겠습니다.

문제 1. 토큰화 패키지를 사용하여 토큰화를 해봅시다.

문제 1-1. 다음 sentence를 문장 단위로 구분하는 문장 토큰화를 실행하세요.(nltk)

```
sentence = 'One of the essential things in the life of a human being is communication. We need to communicate with other human beings to deliver information, express our emotions, present ideas, and much more. The key to communication is language. '
```

문제 1-2. 다음 sentence를 단어 단위로 토큰화해주세요.(nltk)

```
sentence ='We need to communicate with other human beings to deliver information, express our emotions, present ideas, and much more.'
```

```
sentence ='The price\t of burger \nin BurgerKing is Rs.36.\n'
```

HINT1. nltk에는 문장, 구두점, 공백 등을 다루는 토큰화 방식에 따라 여러가지 종류가 있습니다. 적용을 해보며 어떤 차이가 있는지 알아보세요.

문제 1-3. Konlpy(okt) 라이브러리를 사용하여 '1-3.txt' 파일에서 50문장을 골라 한글 형태소 분석을 진행해주세요.

HINT1. 'norm=True, stem=True' 로 설정해주세요.

HINT2. 조사, 어미, 문장 부호는 제외하고 처리해주세요.

문제 2. nltk 라이브러리를 사용하여 토큰화 후 다음 문장에서 불용어 제거를 해보세요.

```
sentence = "We need a common language to communicate, which both ends of the conversation can understand."
```

문제 3. nltk 라이브러리에서는 어간 추출 알고리즘 중 하나인 포터 알고리즘을 제공합니다. 여러 단어들을 넣어보며 원형을 잘 보존하는지 확인해보세요.

문제 3. nltk 라이브러리에는 표제어 추출 알고리즘을 제공합니다. 여러 단어들을 넣어보며 표제어를 잘 추출하는지 확인해보세요.

HINT1. 품사 정보를 추가하면 어근 단어를 더욱 잘 추출합니다.

Chapter 2 임베딩

임베딩은 사람이 사용하는 언어를 컴퓨터가 이해할 수 있는 언어 형태인 벡터로 변환하는 것을 의미합니다. 임베딩을 통해 단어 및 문장 간에 관련성이 어느 정도인지를 계산하며, 문장 안에 담긴 의미/문법적 정보를 함축하여 숫자 형태로 표현합니다.

임베딩 또한 여러 종류가 있는데, 먼저 원-핫 인코딩은 단어가 포함되어 있는 위치에 1 을 넣고 나머지에는 0 값을 채웁니다(딥러닝팀 3 주차 클린업 참고). 하지만 이렇게 하나의 요소만 1 을 갖고 나머지는 모두 0 인 희소벡터인 경우, 단어끼리 관계성(유의/반의어) 없이 독립적인 관계를 갖게 되며, 하나의 단어를 표현하는 데에 코퍼스에 있는 수만큼 차원이 존재하기 때문에 '차원의 저주' 문제가 발생합니다. 따라서 이러한 문제를 해결한 임베딩 방법을 배워보겠습니다.

문제1. 횟수 기반 임베딩 중 하나인 CountVectorizer를 다음 코퍼스에 사용하여 얻은 결과를 배열로 변환해주세요.

```
corpus = [
    'This is the first document.',
    'This is the second second document.',
    'And the third one.',
    'Is this the first document?',
    'The last document?',
]
```

HINT1. stop_words를 사용하여 불용어를 직접 지정한 후 제거하여 진행해주세요.

문제2. 예측 기반 임베딩인 Word2Vec를 사용하여 단어간 의미 유사도를 구해보겠습니다.

문제2-1. '2-1.txt' 파일을 불러와주세요. 파일은 소설 피터팬 원문의 일부분입니다.

문제2-2. 파일 문장을 단어로 모두 토큰화해주세요.

문제2-3. 토큰화된 단어를 모두 소문자로 변환하여 저장해주세요.

문제2-4. CBOW 방식으로 벡터화를 진행한 후 'peter'와 'wendy', 'peter'와 'hook'의 유사도를 각각 구해주세요. (min_count = 1, vector_size = 100, window=5)

문제2-5. 문제 '2-4'를 skip-gram 방식으로 진행해주세요. 설정값도 동일합니다.

→ 데이터 특징, 분석 접근 방법 등에 따라 다른 결과를 보임을 알 수 있습니다. 따라서 상황에 따라서 필요한 라이브러리가 무엇인지 판단하여 사용할 수 있어야 합니다.

(Optional) 문제2-6. Word2Vec의 단점을 보완한 'FastText' 알고리즘을 사용하여 문제 '2-4'를 해결해주세요. 그리고 온라인 텍스트와 같이 오탈자가 많은 경우에 'FastText'가 왜 더욱 잘 작동할지 특징을 간략히만 적어주세요.

문제3. attention은 nlp 분야에서 매우 중요한 매커니즘인데, 이를 사용한 모델 중 대표적인 BERT 모델이 어떻게 토큰화부터 임베딩까지 진행하여 높은 성능을 보여주었는지 알아보겠습니다.

문제3-1. 다음 문장을 사전훈련된 bert tokenizer를 이용하여 토큰화해주세요. 'bert-base-uncased'로 설정하여 기본 모델을 사용하겠습니다.

```
text = 'NLP is an interdisciplinary subfield of linguistics, computer science
, and artificial intelligence'
```

```
{'input_ids': [101, 17953, 2361, 2003, 2019, 18593, 4942, 3790, 1997, 15397, 1010, 3274,
2671, 1010, 1998, 7976, 4454, 102], 'token_type_ids': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0], 'attention_mask': [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1]}
```

위와 같은 결과가 나와야 합니다. 각각 어떤 걸 의미하는지 알아보겠습니다.

문제3-2. Bert는 토큰을 이용하여 문장을 구분하는데 문장 앞에는 '[CLS]', 뒤에는 '[SEP]' 토큰을 붙입니다. 위의 'text' 또한 두 개의 토큰을 앞뒤에 붙여 저장해주세요.

문제3-3. Bert는 tokenize() 함수를 사용하여 토큰화를 진행한 뒤 결과를 확인해주세요. 단어 단위로 토큰화를 한 후 알 수 없는 단어는 더 작은 단위로 분리한 것을 확인할 수 있습니다.

문제3-4. convert_tokens_to_ids() 함수를 사용하여 각 토큰의 인덱스를 알 수 있습니다. 결과값이 위의 'input_ids'인 것을 알 수 있습니다. 결과를 따로 저장해주세요.

참고. 길이가 다른 두 개의 sentence를 입력으로 받은 경우, 'token_type_ids'는 두 문장을 구분하는 역할, 'attention_mask'는 각 문장에서 attention 연산이 필요한 토큰을 표시해주는 역할을 하며 문장의 토큰 길이와 같은 것을 볼 수 있습니다.

문제3-5. 문제에서 하나의 문장을 사용 중이기 때문에 3가지 인풋 모두 사용하지 않아도 됩니다. 'attention_mask'와 같은 리스트만 만들주세요.

문제3-6. 문제 4-4~5에서 만든 두 가지 리스트를 텐서로 변환해주세요.

문제3-7. BertModel(bert-base-uncased, output_hidden_states = True로 설정)를 구축한 후 텐서를 넣어 결과를 확인해주세요.

HINT1. keys()를 사용하여 3가지 아웃풋이 각각 어떤 걸 의미하는지 알 수 있습니다.

Chapter 3 정형데이터 적용(LSTM)

자연어뿐만 아니라 시계열 정보를 가진 정형데이터에도 딥러닝 모델이 적용 가능합니다. 딥러닝팀 클린업 3주차에서 배운 모델 중 LSTM을 간단하게 구현해보겠습니다.

문제1. 'Modelling.csv' 데이터를 불러온 후 구조를 파악해주세요. 이는 기업의 주가 정보 데이터입니다.

문제2. 'Date' 칼럼을 datetime 형식으로 바꾼 후 'Date' 칼럼을 인덱스로 설정해주세요.

문제3. 'Volume' 칼럼을 실수로 변환한 후. 'Volume' 칼럼을 레이블(y), 나머지를 훈련(x) 데이터로 분할해주세요.

문제4. 데이터 간의 분포를 맞추기 위해 스케일링을 해보겠습니다.

문제4-1. StandardScaler를 사용하여 x 데이터를, MinMaxScaler()를 사용하여 y 데이터를 스케일링해주세요.

문제5. x와 y데이터는 253행을 갖고 있는데 그 중 200개는 훈련 데이터셋, 나머지는 테스트 데이터셋으로 각각 분할해주세요.

문제6. 데이터를 모두 텐서로 변환한 후 x 데이터를 LSTM의 입력 형태와 맞추기 위해 형태를 변경해주세요.

문제6-1. Train x 데이터는 [200,1,5], test x 데이터는 [53,1,5]의 형태를 가져야 합니다.

문제7. LSTM 모델을 class의 형태로 만들어 구축해주세요. 모델을 만들 때에는 torch.nn.Module을 상속 받아야하며, __init__, forward 함수가 필수적으로 구현되어 있어야 합니다.

HINT1. __init__ 함수에서는 모델 내의 layer들과 외부 변수를 지정해야 합니다.

HINT2. forward 함수에서 (1) 0으로 초기화한 hidden state를 정의하고 (2) 0으로 초기화한 cell state 를 정의하고 (3) lstm 계층에 hidden state와 cell state를 적용한 후 (4) 마지막 단계의 hidden state를 fully connected layer에 통과시킨 결과를 반환해주세요.

HINT3. 은닉층 뉴런개수는 2, lstm 계층수는 1로 설정해주세요. (이후에 따로 변수로 정의하여 변수명을 써도 괜찮습니다.)

문제8. 알맞은 loss function을 정의하고 optimizer는 Adam(learning rate=0.0001)로 설정해주세요.

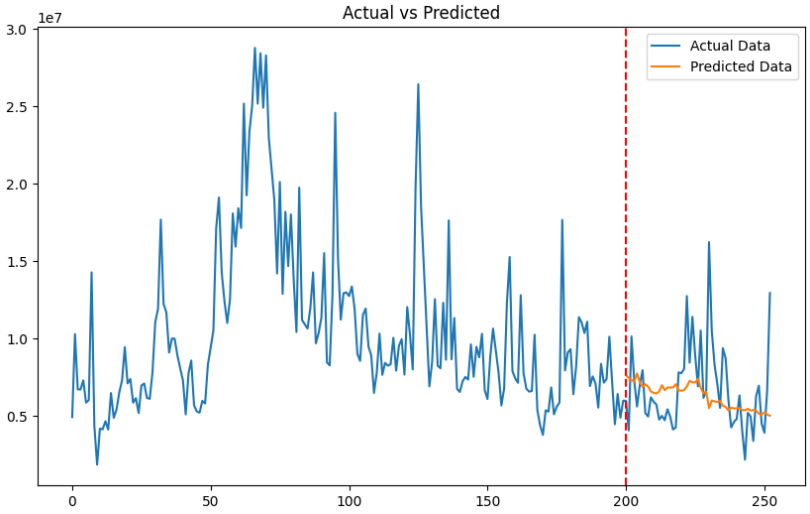
문제9. Epoch 5000회 반복하여 구축한 모델을 학습시켜주세요. Epoch 1000회마다 loss값을 출력하도록 해주세요.

HINT1. 내부에 optimizer의 gradient를 0으로 초기화해주는 코드가 있어야 학습이 가능합니다.

문제10. 학습한 모델로 test data에 적용하여 예측을 진행한 후 plot을 그려주세요.

HINT1. 밑의 plot과 같은 형식으로 만들어주세요.

HINT2. 전처리 과정에서 스케일링을 진행했기 때문에 원래의 값의 범위로 다시 바꿔주어야 합니다.

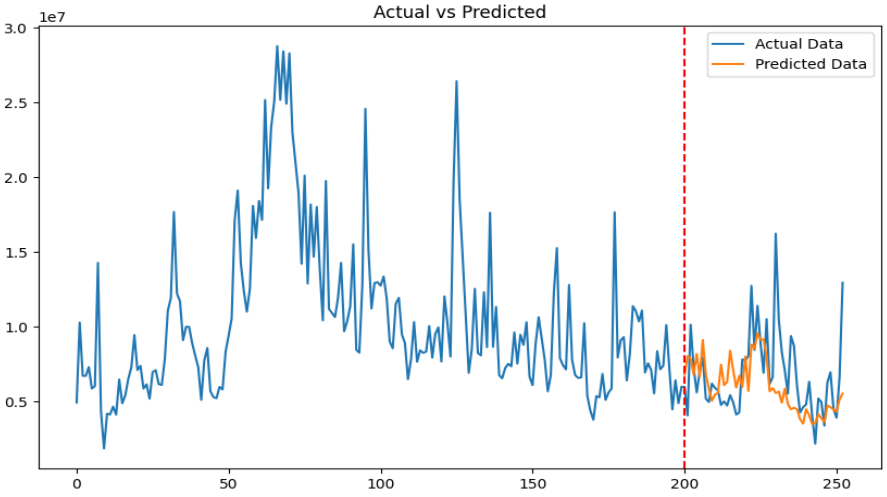


(Optional) 문제11. 지금까지 사용한 lstm의 경우, 이전 시점의 정보로 현재 시점의 정보를 계산하며 과거의 정보만을 사용하였지만 bidirectional lstm은 이후 시점의 데이터를 함께 활용합니다. 어떻게 작동하는지 알아본 후bidirectional lstm으로 test data를 예측한 결과를 똑같이 시각화해주세요.

HINT1. 이전에 사용했던 코드 중 모델 부분만 수정하여 사용하면 됩니다.

HINT2. nn.LSTM()의 'bidirectional' 파라미터를 사용해주세요.

HINT3. Bidirectional lstm은 전방향/역방향을 모두 학습하여 두 개의 계층이 필요하기 때문에 일부 파라미터 입력값이 두 배가 되어야 합니다.



시드에 따라 결과의 차이가 크기 때문에 plot 형태가 같지 않아도 됩니다.