بسمه تعالی

تمرین اول

درس پردازش زبان های طبیعی

سید محمدرضا حسینی

کد دانشجویی: ۴۰۲۲۱۱۶۸۳

دانشکده مهندسی کامپیوتر

مدرس: دکتر عسگری

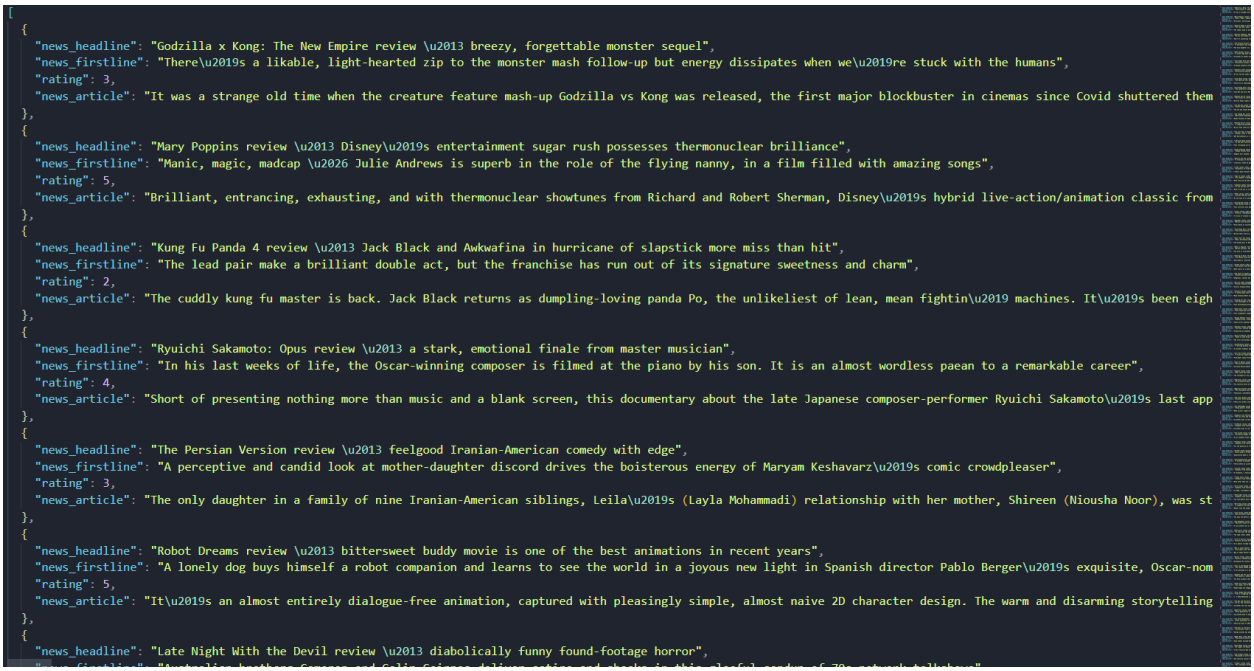
برای این تمرین داده ها را از سایت www.theguardian.com ، crawl کردیم. دادگان جمع آوری شده از بخش نقد فیلم ها میباشد. در ابتدا لینک های موجود در هر صفحه اصلی که لینک مرتبط به نقد هر فیلم میباشد را به دست می آوریم.

```
for i in range(1,300):
    try:
        Main_url = "https://www.theguardian.com/film+tone/reviews?page={0}".format(i)
        page = requests.get(Main_url)
        soup = BeautifulSoup(page.text,"html.parser")
        script_tag = soup.findAll('script', {'type': 'application/ld+json'})
        json_content = json.loads(script_tag[2].string)
        for j in range(0,len(json_content)):
```

سپس از هر لینک اطلاعات را بدست می آوریم. اطلاعات ذخیره شده شامل هدلاین نقد ، امتیاز داده شده به فیلم و متن نقد و یک خط خلاصه نوشته شده میباشد.

```
for j in range(0,len(json_content)):
    url =json_content['itemListElement'][j]['url']
    try:
        news = requests.get(url)
    except:
        df = pd.DataFrame(reviews)
        with open('data.json', 'w') as f:
            json.dump(reviews, f, indent=2)
        df.to_csv("'reviews.csv', index=False")
        continue
    news_soup = BeautifulSoup(news.text, "html.parser")
    paragraphs= news_soup.findAll("p")
    temp =""
    stars = news_soup.find_all('div', class_="dcr-1tlk9ix")
    rating = sum(1 for star in stars if star.find('path').get('fill') == 'currentColor')
    for k in range(1,len(paragraphs)):
        temp+=(paragraphs[k].text)
    news_articles = [{'news_headline': news_soup.find("h1").text,
                      'news_firstline': paragraphs[0].text,
                      'rating': rating,
                      'news_article': temp
                      }
                     ]
    reviews.extend(news_articles)
```

اطلاعات کرال شده را در نهایت در یک فایل json ذخیره سازی میکنیم.



پس از اینکار به سراغ پردازش دادگان میرویم. در بخش اول تسک انتخاب شده به دست آوردن عبارات کلیدی و صفات استفاده شده در این نقد ها میباشد. مراحل مرتبط به کد ها در نوتبوک به شکل مناسب کامنت گذاری شده است.

برای اینکار باید پیش پردازش های گفته شده را انجام دهیم تا انجام اینکار راحت تر شود و اطلاعات با ارزش تر باقی بمانند. در ابتدا تمامی Punctuation ها را حذف میکنیم. سپس تمامی کلمات را Lowercase کرده و در نهایت ‘s های موجود در هر کلمه را حذف میکنیم. برای نشان دادن عملکرد از Porter stemmer استفاده کردیم. با استفاده از stemming اتفاقی که می افتد این است که پس از دادن دادگان به Spacy برای شناسایی صفات و عبارات کلیدی این مدل به علت stemming اتفاق افتاده دچار مشکل میشود و نمیتواند این کار را به شکل درست انجام دهد. به عنوان مثال کلمه strange در هنگام stemming به strang تبدیل میشود و آن را verb تشخیص میدهد که این اشتباه با توجه به اینکه تسک ما تشخیص بیشترین صفات به کار رفته در تحلیل فیلم ها میباشد غلط میباشد.

```
1  sentences_a = [sent for sent in doc_preproccessed[0].sents]
2  for x in sentences_a:
3      print([(tok.pos_,tok.lemma_) for tok in x])
```

```
[('SPACE', ' '), ('VERB', 'strang'), ('ADJ', 'old'), ('NOUN', 'time'), ('NOUN',
'creatur'), ('NOUN', 'featur'), ('NOUN', 'mash'), ('NOUN', 'godzilla'), ('ADP', 'vs'),
('PROPN', 'kong'), ('VERB', 'relea'), ('ADV', 'first'), ('ADJ', 'major'), ('PROPN',
'blockbust'), ('PROPN', 'cinema'), ('PROPN', 'sinc'), ('PROPN', 'covid'), ('PROPN',
```

```
sents = [sent for sent in doc[0].sents]
for x in sents:
    print([(tok.pos_,tok.lemma_) for tok in x])
```

```
[('PRON', 'it'), ('AUX', 'be'), ('DET', 'a'), ('ADJ', 'strange'), ('ADJ', 'old'),
('NOUN', 'time'), ('SCONJ', 'when'), ('DET', 'the'), ('NOUN', 'creature'), ('VERB',
'feature'), ('NOUN', 'mash'), ('PUNCT', '-'), ('NOUN', 'up'), ('PROPN', 'Godzilla'),
('ADP', 'vs'), ('PROPN', 'Kong'), ('AUX', 'be'), ('VERB', 'release'), ('PUNCT', ','),
('DET', 'the'), ('ADJ', 'first'), ('ADJ', 'major'), ('NOUN', 'blockbuster'), ('ADP',
'in'), ('NOUN', 'cinemas'), ('SCONJ', 'since'), ('PROPN', 'Covid'), ('VERB', 'shutter')
, ('PRON', 'they'), ('DET', 'all'), ('DET', 'a'), ('NOUN', 'year'), ('ADV', 'prior'),
('PUNCT', '.')]
```

اطلاعات مرتبط با دیتای استفاده شده بعد از پیش پردازش گفته شده در بالا. این پیش پردازش شامل stemming نمیباشد.

```
import itertools
sentence_words = list(itertools.chain(*tokenized_reviews))
print ('%-16s' % 'Number of words', '%-16s' % len(sentence_words))
print ('%-16s' % 'Number of unique words', '%-16s' % len(set(sentence_words)))
avg=np.sum([len(word) for word in sentence_words])/len(sentence_words)
print ('%-16s' % 'Average word length', '%-16s' % avg)
avg_sentence_length_c=np.mean([len(' '.join(sentence)) for sentence in sentence_words])
print ('%-16s' % 'Average sentence length in characters', '%-16s' % avg_sentence_length_c)
```

```
Number of words  327502
Number of unique words 39178
Average word length 6.097312382825143
Average sentence length in characters 11.194624765650286
```

با مقایسه داده ای که این پیش پردازش ها بر روی آن انجام شده در مقابل داده ای که فقط جملات توکنایز شده اند مشخص میشود با توجه به اینکه کلمات حذف شده تاثیری در تسک گفته شده ندارن، با حذف آن ها میتواند اطلاعات را سریعتر و به صورت موثرتری بدست آورد.

همانطور که در تصویر زیر مشخص است توانسته ایم تمامی Noun_chuncks ها را بدست آوریم.

```
keywords = []
for sentence in doc:
    temp=[]
    for chunk in sentence.noun_chunks:
        if chunk.text.lower() not in nlp.Defaults.stop_words:
            temp.append(chunk.text)
    keywords.append(temp)
```

```
np.array(keywords)
```

```
C:\Users\smrh1\AppData\Local\Temp\ipykernel_34188\2185074432.py:1: VisibleDeprecationWarni
different lengths or shapes) is deprecated. If you meant to do this, you must specify 'dty
  np.array(keywords)
```

	0
3	[music, a blank screen, this documentar...
4	[The only daughter, a family, nine Iran...
5	[an almost entirely dialogue-free anima...
6	[A malign presence, the corner, poison,...
7	[all the hot takes, the cinema, the sug...
8	[no leprechauns, this abysmal romantic ...

پس از آن بیشترین صفات استفاده شده توسط نویسندگان در این نقد ها را بدست آوردیم.

```
adj_freqdist = FreqDist(itertools.chain(*adjectives))
top40words=adj_freqdist.most_common(40)      # show the top 20 (word, frequency) pairs
print ('%-16s' % 'word', '%-16s' % 'Frequency','%-16s' %  '% of the total')
for topword in top40words:
    percent=(topword[1]/len(tokenized_reviews))
    print ('%-16s' % topword[0], '%-16s' % topword[1],'%-16s' %  percent)
```

word	Frequency	% of the total
old	613	0.5125418060200669
new	586	0.4899665551839465
good	561	0.4690635451505017
own	549	0.4590301003344482
young	544	0.45484949832775917
more	538	0.4498327759197324
other	513	0.4289297658862876
first	509	0.42558528428093645
-	491	0.4105351170568562
little	441	0.36872909698996653

در بخش دوم با بررسی هدلاین های هر نقد با توجه به پترن موجود در آن سعی میکنیم که اسم فیلم را استخراج کنیم.

با توجه به اینکه به دنبال اسم فیلم هستیم پیش پردازش خاصی را مد نظر نداریم چون در صورت حذف کردن حتی stopwords اسم فیلم ممکن است متفاوت شود. تنها کاری که میکنیم توکنایز کردن جملات است. سپس با استفاده از regex پترن مدنظر را وارد کرده و سعی میکنیم اسم فیلم را بدست آوریم.

همانطور که در تصویر زیر مشخص است با استفاده از ابزار پردازش متن توانسته ایم با دقت مناسبی اسامی فیلم ها را از هدلاین نقد ها استخراج کنیم. این اسامی میتواند برای بهتر و کامل کردن دادگان استفاده شده و سبب شود که به ازای اسم فیلم ما نقد نوشته شده را نگه داری کنیم.

```
import re
movie_names=[]
for i in movie_nouns:
    for j,item in enumerate(i):
        # Use regular expression to find the movie name before the word "review"
        match = re.search(r"^(.*?)\s*review", item, re.IGNORECASE)
        if match:
            if len(i[:j])>0:
                temp=((i[:j].pop().strip())+" " + match.group(1).strip())
            else:
                temp= match.group(1).strip()
            movie_names.append([temp])
np.array(movie_names)
```

	0
0	godzilla x kong the new empire
1	mary poppins
2	kung fu panda 4
3	ryuichi sakamoto opus
4	the persian version
5	the devil