درس پردازش زبان های طبیعی

تمرین دوم

استخراج ویژگی های کالا از نظرات

ترم اول سال ۱۴۰۱

استاد محترم :

دکتر احسان الدین عسگری

اعضای گروه سوم :

حمیدرضا امیرزاده گوغری

محمد حسین سامتی

حسن صبور

برای استخراج ویژگی ها کالا ابتدا ویژگی های عمومی یک کالا مانند قیمت و اندازه و کیفیت را مورد بررسی کردیم و برای بررسی ویژگی ها به دسته بندی کالا های خاص رفتیم مثلا در مورد کتاب نوع جلد و قطع و حجم کتاب و جنس کاغذ و .. و یا در مورد لباس جنس و کیفیت دوخت و یا در مورد کالا های دیجیتال مثل نوع باتری و شارژدهی و سرعت لپ تاب و هارد و فلش و .. پرداختیم .

ابتدا پیش پردازشی روی جملات و تقسیم متن نظرات به جملات را انجام دادیم و سپس جملات یک نظر را به همه ی توابع برای بررسی ویژگی ها دادیم و نهایتا خروجی را نمایش دادیم .

```
def preprocess(comment):
    # Sentence tokenization
    comment_sents = sent_tokenize(comment)

    # Remove punctuations
    comment_sents_nopunc = [rem_punc(sent) for sent in comment_sents]

    # Word tokenization
    tokens = [[word_tokenize(word) for word in sent.split()] for sent in comment_sents_nopunc]

    # Normalization
    tokens_norm  = [[normalizer.normalize(token[0]) for token in ls] for ls in tokens]

    # Lemmatization
    tokens_lemm = [[lemmatizer.lemmatize(token) for token in ls] for ls in tokens_norm]

    # Removing stopwords
    tokens_nostop = [[token for token in ls if token not in stopwords] for ls in tokens_lemm]

    return tokens_nostop
```

برای نمونه در تابع بالا پیش پردازشی روی نظر انجام می شود .

یک تابع نیز برای تشخیص منفی و یا مثبت فعل جمله را تشخیص دهد .

```
def Is_Negative_Verb(verb):
    lem_verb = lemmatizer.lemmatize(normalizer.normalize(verb))
    if verb.startswith('ن') and not lem_verb.startswith('ن') :
        return True
    if verb == 'نیست':
        return True
    return False
```

حالا برای تشخیص هر ویژگی یک تابع برای بررسی آن نوشتیم برای نمونه :

```
def cost_extract(sentences, phrases):
    """ Output : one of three classes ['کم' ,'مناسب' ,'زیاد'] """

    for i,s in enumerate(sentences):
        for key in phrases:
            span = re.search(key,s)
            if span:
                start = span.start()
                sent = sentences[i][start:]

                # neg_sent = False
                tags = tagger.tag(sent.split())
                for tag in tags:
                    if tag[1] == 'V':
                        verb = tag[0]
                        neg_sent = Is_Negative_Verb(verb)
                        # if verb[0] == 'ن':
                        #     neg_sent = True

                pattern_ziad = r'زیا+د | بالا+ | نگرا+ | نگرو+ | بد'
                if re.search(pattern_ziad, sent):
                    if neg_sent == False:
                        return {'زیاد' : 'قیمت'}
                    else:
                        return {'مناسب' : 'قیمت'}

                pattern_monaseb = r'منا+سب | به صرفه | منطقی | متناسب | خوب'
                if re.search(pattern_monaseb, sent):
                    if neg_sent == False:
                        return {'مناسب' : 'قیمت'}
                    else:
                        return {'زیاد' : 'قیمت'}

                pattern_kam = r'کم | ارزا+ن | مفت | پایی+ن'
                if re.search(pattern_kam, sent):
```

بعد از نوشتن قانون ها برای تشخیص ویژگی ها سعی کردیم در تابع به ران مربوط به هرکلاس این متد ها را فراخوانی کنیم .

```python
class CLOTHES_PROPERTIES_EXTRACTOR:
    def __init__(self, text):
        self.normalizer = Normalizer()
        self.lemmatizer = Lemmatizer()
        self.stemmer = Stemmer()
        self.tagger = POSTagger(model='postagger.model')
        self.text = text

    def run(self, flag=True):
        dic = {}
        dic = dic | self.SIZE_(flag)
        dic = dic | self.MATERIAL_(flag)
        dic = dic | self.QUALITY_(flag)
        dic = dic | self.MODEL_(flag)
        dic = dic | self.DESIGN_(flag)
        return dic
```

حالا با یک کلاس اصلی ۳ کلاس فرعی را یک شی ساخته و صدا می زنیم .

```python
class ProductCharacteristicsExteractor:
    def __init__(self,text):
        self.text=text

    def run(self):
  #       GeneralPropertiesExtrator()
        properties={}
        b=BookPropertiesExtrator()

        s=CLOTHES_PROPERTIES_EXTRACTOR(self.text)
        Digital=DIGITAL_PROPERTIES_EXTRACTOR(self.text)
        properties.update(Digital.run())
        properties.update(s.run())
        properties.update(b.run(self.text))
        return properties
```

بعد از انجام این مراحل برای تست سرعت و صحت این ماژول یک فایل تست نوشتیم که یک فایل json را فراخوانی که ورودی را به ماژول و خروجی ها را با هم مقایسه می کند . با بررسی تعداد موارد صحیح دقت را حساب می کند و برمی گرداند .

```python
from ProductCharacteristicsExteractor import *
import json
class Test:
    def runTest():
        fp=open('Testsamples.json', 'r')
        obj = json.load(fp)
        count=len(obj)
        correct=0
        for i in obj:
            t=ProductCharacteristicsExteractor(i["inputcomment"])
            if t==i["output"]:
                correct=correct+1
        return correct/count
```

```
[
    {
        "inputcomment":"کلاسه بودنش بد بود ولی چون جلدش شومیز بود خوب بود",
        "output":"{'نوع جلد' :'شومیز' ,'جنس' :'کلاسه'}"
    },
    {
        "inputcomment":"مشخص نبود که قیمت کالا خیلی بالا است. بنابراین، ارزش نداشت",
        "output":"{'قیمت' :'زیاد'}"
    },
    {
        "inputcomment":"قیمت مناسب بود اما چون تخفیف دارد بد نبود",
        "output":"{'قیمت' :'زیاد' ,'تخفیف' :'دارد'}"
    },
    {
        "inputcomment":"هزینه ای که پرداختیم خیلی بالا بود و کار خوب میکند",
        "output":"{'کارکرد' :'خوب' ,'قیمت' :'زیاد'}"
    },
    {
        "inputcomment":"سرعت خوبی دارد",
        "output":"{'سرعت'] :'سریع']}"
    },
    {
        "inputcomment":"باتری تا مدت طولانی شارژنگه می دارد",
        "output":"{'باتری'] :'خوب']}"
    },
    {
        "inputcomment":"کلاسه بودنش بد بود ولی چون جلدش شومیز بود خوب بود",
        "output":"{'نوع جلد' :'شومیز' ,'جنس' :'کلاسه'}"
    }
]
```

این ماژول با افزوده شدن ویژگی های زیاد و قانون های جدید می تواند تکمیل تر شده و در اختیار بقیه دوستان عزیز قرار گیرد .

با تشکر