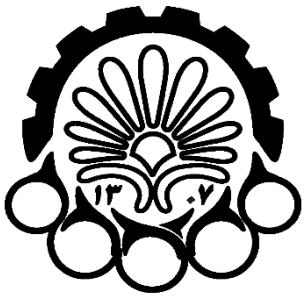


فاز اول پروژه بازیابی اطلاعات

نام دانشجو: سپهر عسگریان ابیانه

شماره دانشجو: 9531901

اردیبهشت 1399

باسمه تعالی

مالی‌شرکتی

نام و نام خانوادگی: سپهر عسگریان **رشته تحصیلی: مهندسی کامپیوتر** **دانشگاه صنعتی امیرکبیر**

گزارش پروژه

ابتدا برای هر دو حالت تمامی دیتا ها با یک دیگر مرج شده اند و تشکیل دیتایی با 55000 داده را داده اند.

ابتدا حالت اول به اسم فایل حالت اول که در jupyter پیاده سازی شده است و همچنین نتایج آن نیز در آن فایل قابل مشاهده است .

دیتای اولیه به صورت زیر قابل مشاهده است

```
In [134]: data=df
          data['content']

Out[134]: 0       <strong><span dir="RTL">به گزارش <a href=...
          1       <p>کاهش تنوع مدارس، شعار سال‌های اخیر سکاندارا...
          2       <div class="hideTag">\n <a href="/fa/service/3...
          3       <div class="item-text" itemprop="articleBody">...
          4       <p style="text-align: justify;">به گزارش <a cl...
          5       <div class="body col-xs-36"> \n <a class="ente...
          6       <div class="item-text" itemprop="articleBody">...
          7       <div class="hideTag">\n <a href="/fa/service/3...
          8       <div class="image-news pull-right mrxxl image-...
          9       <p>به گزارش ایسنا، پس از اینکه سمیه یزدانی در ...
          10      <p><img class="image_btn" style="margin: 10px ...
          11      <a href="/" class="irinn_link">به گزارش پایگاه...
          12      <div dir="rtl">\n به گزارش خبرنگار \n <span st...
          13      <div class="hideTag">\n <a href="/fa/service/8...
          14      <p style="text-align:justify"><span style="col...
          15      <p>ایران به واسطه عدم انجام تعهدات برجامی از س...
          16      به گزارش خبرنگار <a href="http://shabesta...
          17      <p><img class="image_btn" style="margin: 10px ...
          18      <div dir="rtl">\n به گزارش تابناک به نقل از &nb...
          19      <p>استرس تنها فاکتوری نیست که موجب سفید شدن زو...
          20      <img align="left" class="news_corner_image" sr...
          21      <p><img class="image_btn" style="margin: 10px ...
          22      <p><img class="image_btn" style="margin: 0px 8...
          23      <div dir="rtl">\n حساب کاربری زندانیان سیاسی ع...
          24      <p><span style="font-size: 14pt;"><img class="...
          25      <p>به گزارش خبرنگار<span style="font-size: 12p...
          26      <div class="hideTag">\n <a href="/fa/service/6...
          27      <div class="hideTag">\n <a href="/fa/service/3...
          28      <div class="hideTag">\n <a href="/fa/service/6...
          29      <div class="hideTag">\n <a href="/fa/service/8...
                                          ...
          7714    <div class="row " id="slideshowDisplay"> \n <d...
          7715    <p>به گزارش ایسنا، به نقل از سایت شبکه روسیا ا...
```

1. **حالت اول**

در حالت اول نرمالایز خیلی ساده همراه با حذف stop words داشتیم به همره توکنایز کردن بسیار ساده که مثلا برای واژه کتاب ها توکنایز به صورت کتاب و ها تشکیل میشد که در اینجا توکنایز اضافی داریم . ابتدا در این بخش با استفاده از دستور

```
In [83]: regex = re.compile('[a-zA-Z]')
         data['content'] = data['content'].map(lambda x: re.sub('[a-zA-Z]', '', x))
```

باسمه تعالی

مالی‌شرکتی

نام و نام خانوادگی: سپهر عسگریان رشته تحصیلی: مهندسی کامپیوتر دانشگاه صنعتی امیرکبیر

کلیه واژگان انگلیسی را حذف کرده همچنین با استفاده از دستورات زیر علائم نگارشی ، تگ های html اعداد و اموجی ها همگی حذف گردیده اند . در اینجا از دیتور re.sub استفاد شده است و از کتابخانه regex بهره برده شده است . برای حذف علائم نگارشی همانطور که در کد نیز قابل مشاهده است با یک فاصله در متن عوض گردیده است

In [81]:
```
import string
import re
data['content'] = data['content'].map(lambda x: re.sub(r'&',' ', x))
data['content'] = data['content'].map(lambda x: re.sub('<[^>]+>', '', x))
data['content'] = data['content'].map(lambda x: re.sub(' href\s*=\s*\"[^\"]*', '', x))
regex = re.compile('[a-zA-Z]')


data['content'] = data['content'].map(lambda x: re.sub('((www\.[^\s]+)|(https?://[^\s]+))', 'URL', x)) # remove URLs
data['content'] = data['content'].map(lambda x: re.sub('@[^\s]+', 'AT_USER', x))
data['content'] = data['content'].map(lambda x: re.sub(r'#([^\s]+)', r'\1', x))
def remove_punct(text):
    text = re.sub('[0-9]+', '', text)
    text = re.sub('[\'-ٔ]+', '', text)

    return text
print("done1")
data['content']= data['content'].apply(lambda x: remove_punct(x))

#data['content'] = data['content'].map(lambda x: re.sub(r'(',' ', x))
#data['content'] = data['content'].map(lambda x: re.sub(r')',' ', x))



#for i in range(len(string.punctuation)):
#    print(i)
#    data['content']=punjanction(data['content'],i)

data['content']=data['content'].str.replace('[{}]'.format(string.punctuation), '')

clean = re.compile('<.*?>')
data['content'] = data['content'].map(lambda x: re.sub(clean,'', x))

def emoji(text):
    import emoji

    allchars = [str for str in text.decode('utf-8')]
    emoji_list = [c for c in allchars if c in emoji.UNICODE_EMOJI]
    clean_text = ' '.join([str for str in text.decode('utf-8').split() if not any(i in str for i in emoji_list)])
    return clean_text
```

حال داده را بعد از اینکه یک نرمالایز ساده شد در زیر نمایش داده شده است برای مثال خبر اول آمده است که فاقد هرگونه کلمه انگلیسی یا تگ میباشد. لازم به ذکر است نرمالایز کامل در مرحله دوم دقیقا طبق صورت پروژه انجام شده است

In [138]:
```
data['content'][2]
```

Out[138]:
```
'خبار ورزشی صادق ورمزیار در گفت وگو با خبرنگار ورزشی خبرگزاری تسنیم، درباره برتری بر صفر تیم فوتبال استقلال مقابل فجر سپاسی شیراز در مرحله یک هشتم نهایی جام حذفٔ
ی اظهار داشت این مسابقه نتیجه خوبی برای استقلال در پی داشت و صعود به مرحله بعدی جام حذفی، نکته بسیار خوبی بود، اما تیم از لحاظ فنی خیلی قوی نبود و هنوز به زمان و دَ
مرین بیشتری نیاز دارد استقلال باید بیشتر از این کار کند تا در لیگ برتر موفق باشد ، وی ضمن بیان این جمله تصریح کرد در نیمه اول چیزی از استقلال ندیدیم در نیمه دوم اما مالٔ
.کیت توپ و میدان را به خوبی در اختیار داشتیم، با این حال نتوانستیم به خوبی موقعیت های گلزنی طراحی کنیم معلوم نبود که تیم می خواهد از چه راهی به گل برسد، از ارسال از
جناحین یا پاس در عمق این حملات سردرگم باعث شد استقلال زمان را از دست بدهد استراماجونی یک قدم رو به جلو برداشتیم اما هنوز خیلی کار داریم پیشکسوت باشگاه استقلال در یا
سخ به این پرسش که آیا آبی پوشان به بازیکنی جدید نیاز دارند یا خیر، خاطرنشان کرد استقلال به هافبکی طراح نیاز دارد تا به بازی نظم دهد و حملات تیم را طراحی کند استقلال هم
نین به زمان احتیاج دارد تا هماهنگ شود و افکار استراماجونی را در تیم ببینیم جابه جایی متعدد بازیکنان و کمبود زمان به بازیکنان اجازه نداده که به هماهنگی برسند استقلال شدیداً به
چند بازیکنی که بتوانند تیم را راه بیندازند، نیاز دارد ورمزیار با اشاره به تفاوت آبی پوشان با روزهای اوج شان به تسنیم گفت تیم فعلی با استقلال واقعی خیلی فاصله دارد و بازیکنان با
ید بدانند پیراهن چه تیمی را پوشیده اند استقلال میلیون ها هوادار دارد که آنها منتظر بازی های خوب این بازیکنان هستند این طرفداران با بازی های استقلال زندگی می کنند و بازیکنان
باید این مسائل را بدانند وی در پاسخ به این پرسش که آیا عملکرد شاگردان استراماجونی تا اینجای فصل، امیدوارکننده بوده است یا خیر، بیان کرد شکی در این نیست که استقلال از ابتدا
ی فصل تاکنون، سیری صعودی طی کرده است پس از اینکه در یکی دو بازی عملکرد متوسطی داشتیم، کم کم به درجه خوبی از اعتمادبه نفس رسیدیم و توانستیم سه بازی متوالی را ببٔ
ریم استقلال برای رسیدن به تیم آرمانی فاصله زیادی دارد، اما این سه پیروزی می تواند نقطه امیدی برای تیم و هواداران باشد مربی پیشین تیم فوتبال استقلال درباره شانس موفقیت این تیٔ
م در هفته های آتی لیگ برتر فوتبال اظهار داشت استقلال در لیگ برتر، بازی های سختی پیش رو دارد که فکر می کنم اگر پیش از این دیدارها به هماهنگی لازم برسد، می توان به
ِفقیت تیم امیدوار بود، اما اگر این هماهنگی تا بازی های سخت هم ایجاد نشود و روند پیروزی های استقلال متوقف شود، تیم دچار استرس می شود که این برای استقلال خطرناک است
رمزیار درباره شعارهای هواداران استقلال در دیدار این تیم برابر فجر سپاسی علیه امیرحسین فتحی، مدیرعامل باشگاه گفت فکر نمی کنم مدیران تا تیم فصل بتوانند کاری انجام دهند قطعاً
```

حال در این مرحله تمامی **stop words**ها از یک فایل **word** که در اینترنت نیز موجود بود برداشته شده است و خوانده شده

سپس همانطور که در صورت پروژه آمده است **toekenize** بر اساس فاصله ای که موجود است در متن انتخاب شده است و همچنین **stopwords**ها از متن بیرون کشیده شده اند(فایل **stopwords** استفاده شده در داکیومنت گزاشته شده است) همانند شکل زیر قابل مشاهده است.

In [37]:
```
#Read stopwords file .txt
def read_file_stopwords(addrs):#read file
    list_result = []
    with open(addrs, 'r', encoding="utf_8") as f:
        stopwords = f.read()
        stopwords = stopwords.splitlines()
        for word in stopwords:
            list_result.append(word)

    return list_result
stopword=read_file_stopwords(r'C:\Users\sepehr\Desktop\IR\project\IR-S19-project-data (1)\IR-S19-project-data\project1_phase1
```

In [38]:
```
#Tokenization with space

def tokenization(text):
    text = re.split('\W+', text)
    return text

data['content'] = data['content'].apply(lambda x: tokenization(x))
print("tokenization done")

#Removing stop words after Tokenization
def remove_stopwords(text):#remove stop words
    text = [word for word in text if word not in stopword]
    return text

data['content'] = data['content'].apply(lambda x: remove_stopwords(x))
print("remove_stopwords done")
```

```
tokenization done
remove_stopwords done
```

حال برای مثال بعد از مراحل بالا به شکل زیر میشود خروجی

```
In [210]: data['content'][3445]

Out[210]: ['حیدر',
           'عباس',
           'زاده',
           'فرمانده',
           'نیروی',
           'انتظامی',
           'استان',
           'خوزستان',
           'امروز',
           'فارس',
           'مرز',
           'تلمجه',
           'اظهار',
           'راستای',
           'تامین',
           'امنیت',
           'زائران',
           'اربعین',
           'حسینی',
```

2-حالت دوم :

در حالت دوم نرمالایز کردن با دقت بیشتری انجام میشود همانطور که در صورت پروژه نیز گفته شده است پس از مرج کردن دیتا ابتدا حالت های مختلف کلمه به یک حالت واحد تبدیل شده است

کد آن مانند شکل زیر میباشد

```
def cheak_spell_arabic(text):
    aftery = re.sub("ئ" ,"ء", text)
    aftera = re.sub(r"[أٱإآ]", r"ا", aftery)
    abfterb = re.sub(r"[بـبـ]", r"ب", aftera)
    afterp = re.sub(r"[پـپـ]", r"پ", abfterb)
    afterт = re.sub(r"[تثٹـتـثـ]", r"ت", afterp)
    afterc = re.sub(r"[ثـثـ]", r"ث", afterт)
    afterj = re.sub(r"[ججـجـ]", r"ج", afterc)
    afterch = re.sub(r"[چچـ]", r"چ", afterj)
    afterh = re.sub(r"[ححـخـ]", r"ح", afterch)
    afterkh = re.sub(r"[خخـ]", r"خ", afterh)
    afterd = re.sub(r"[ڈدد]", r"د", afterkh)
    afterz = re.sub(r"[ذذ]", r"ذ", afterd)
    afterr = re.sub(r"[ڑزڑرر]", r"ر", afterz)
    afterzi = re.sub(r"[زز]", r"ز", afterr)
    afterzh = re.sub(r"ژ", r"ژ", afterzi)
```

برای مثال برای حلت زیر خروجی به صورت زیر میباشد

```
print(cheak_spell_arabic('آآآكككككككككك'))
```

```
آآآكككككككككك
```

حال علاوه بر آن انواع علائم تشدید تنوین کسره و غیره با کد زیر و مثال زیر حذف شده است

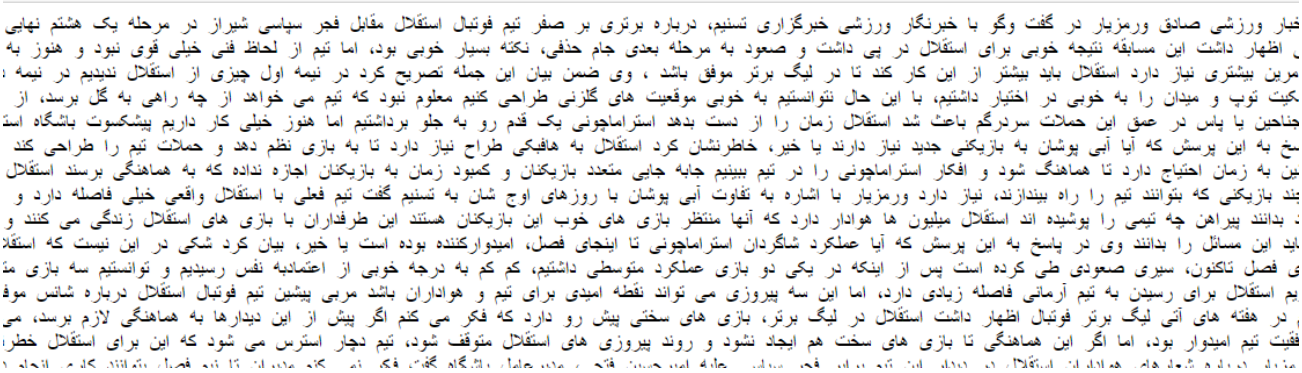
```
def removetashdid(text):
     text = re.sub('\u0651', '', text)#tashdid
     text = re.sub('\u064a', '', text) #yeh
     text = re.sub('\u0649', '', text) #yeh
     text = re.sub('\u0652', '', text) #sukon
     text = re.sub('\u064b', '', text) #fathatan
     text = re.sub('\u064e', '', text) #fatha
     text = re.sub('\u0650', '', text) #kasra
     text = re.sub('\xa0',  ' ', text) #bad spaces

     return text

data['content']= data['content'].apply(lambda x: removetashdid(x))
print(removetashdid('یاسّر گر دیده است در زمان مناسبش باشدک '))
```

```
یاسر گر دیده است در زمان مناسبش باشدک
```

همانطور که دیده میشود تشدید در زیر حذف گردیده است

بقیه ی نرمالایز همانند حالت اول انجام شده است مانند حذف اعداد ، حذف اموجی، حذف علائم نگارشی و غیره که در شکل حالت اول نیز موجود میباشد

بعد از نرمالایز خروجی ها به صورت زیر میباشند



حال در این قسمت علاوه بر خواندن فایل **stopwords** فایلی به نام عبارت ها نیز ذخیره شده است که از گیت‌هاب فایل ورودی آن را پیدا کرده ام. این فایل شامل عباراتی مثل فیه‌مافیه معذلک علی ای حال و غیره میباشد که همه در توکن کردن به یک شکل میباشند پس بنابراین برای اینکار این عبارات خوانده شده است و همانند شکل زیر مشکل آن هندل شده است

```
def _fix_phrases(text):
    for phrase in phrases:
        text = text.replace(phrase, iget_whole_phrase(phrase))
        text = _fix_white_space(text)
    return text


def iget_whole_phrase(ph):
    ph = ph.replace(" ", '\u200c')
    return ph

def _fix_white_space(text):
    text = re.sub("\s+", " ", text)
    text = text.strip()
    return text
```

که در اینجا برای هرکدام از آن ها بینشان اگر فاصله ا موجود میباشد پاک شده است و برای مثال علی ای حال به صورت جدا آمده که همانطور که می‌بینید به صورت سر هم نوشته شده است

همچنین این عبارت

```
data['content'] = data['content'].apply(lambda x: fix_phrases(x))
```

به معنای این استکه هر **text** آن را میفرستد در تابع و خروجی می‌گیرد

سپس برای اینکه علائم جمع توکن اضافی همانند حالت اول به حساب نگیرد کد زیر نوشته شده است که علاوه بر علائم جمع پسوند ها و پیوند های مهم دیگر نیز دقت شده که در صورت مشاهده با نیم فاصله بیایند

```
def space_handel(string):   # space to half space
    string = re.sub(r'^(نمی|می|بی)( )', r'\1•', string)
    #print(string)
    string = re.sub(r'( )(نمی|می|بی)( )', r'\1\2•', string)
    string = re.sub(r'( )(ها|های|ایی|هایم|هایت|هایش|هایمان|هایتان|هایشان|ات|ان|ین)'
              r'|(انی|بان|ام|ای|یم|ید|اید|اند|بودم|بودی|بود|بودیم|بودید|بودند|ست)( )', r'•\2\3', string)
   # print(string)
    string = re.sub(r'( )(شده|نشده)( )', r'•\2•', string)
    #print(string)
    string = re.sub(r'( )(ها|های|گر|گرایی|گرایان|شناس|شناسی|گذاری|گذار|گذاران|شناسان|گیری|آوری|سازی|طلب|طلبان'
              r'|بندی|کننده|کنندگان|پرداز|پردازی|پردازان|آمیز|سنجی|ریزی|داری|دهنده|پذیری'
              r'|بند|رسانی|دهندگان|دار|پذیر|پذیران|گر|ریز|یاب|یابی|گانه|گانه‌ای|انگاری|گا)( )?', r'•\2\3 ', string)

    return string
data['content'] = data['content'].apply(lambda x: space_handel(x))
```

برا مثال در شکل زیر این تابع برای رشته مشخص شده پیوشند ها و پسوند ها را درست کرده است

حال مانند قبل **tokenize**و **stopwords** ها هندل شده اند لازم به ذکر است که در **TOKENIZE** علائم پیشوندی و پسوندی به صورت کامل درست جدا شده اند و مزیت آن این است که در مرحله ریشه یابی با استفاده از کتابخانه **hazm** به خوبی ریشه یابی و **lemmatize** انجام میشود

سپس با استفاده از کتابخوانه **hazm** ریشه یابی و **lemmatize** انجام دشه است همانطور که در شکل زیر میبینید

موارد خواسته شده بدست آمده شده است

```
from hazm import *
lemmatizer = Lemmatizer()
print(lemmatizer.lemmatize('گفت').split('#')[1])
print(lemmatizer.lemmatize('گو'))
print(lemmatizer.lemmatize('رود'))
print(lemmatizer.lemmatize('رو'))
print(lemmatizer.lemmatize('خواه'))
print(lemmatizer.lemmatize('سپاس'))
print(lemmatizer.lemmatize('هنر'))
print(lemmatizer.lemmatize('شریف'))
print(lemmatizer.lemmatize('دوست'))
print(lemmatizer.lemmatize('یاد'))
print(lemmatizer.lemmatize('توان'))
print(lemmatizer.lemmatize('شنو') )
print(lemmatizer.lemmatize('کرد'))
print(lemmatizer.lemmatize('ساز') )
print(lemmatizer.lemmatize('دان')  )
```

```
گو
گو
رود
رو
خواه
سپاس
هنر
شریف
دوست
یاد
توان
شنو
کرد#کن
ساز
دان
```

```
stemmer = Stemmer()
stemmer.stem('کتاب‌ها')
```

```
'کتاب'
```

این عملیات مانند عملیات ریشه یابی برای انگلیسی است با این تفاوت که library آن hazm میباشد

```
stemmer = Stemmer()
lemmatiz = Lemmatizer()

print("done3")
def stemming(text):
    text = [stemmer.stem(word) for word in text]
    return text

data['Tweet_stemmed'] = data['content'].apply(lambda x: stemming(x))


print("done4")
def lemmatizer(text):
    #print("new Line")
    s='#'
    listof=[]
    for word in text:
      lemi=  lemmatiz.lemmatize(word)
      if(s in lemi):
            lemi=lemi.split("#")[1]


      listof.append(str(lemi))
    return listof

data['Tweet_lemmatized'] = data['Tweet_stemmed'].apply(lambda x: lemmatizer(x))
```

```
done3
done4
```

نمونه خروجی :

```
In [208]: data['Tweet_lemmatized'][0]

Out[208]: ['گزار',
           'خبرگزار',
           'تنبس',
           'نقل',
           'آژانس',
           'خبر',
           'پاخ',
           'قول',
           'اردو',
           'اتل',
           'جنوب',
           'اقتانس',
           'نشر',
           'خبرنامه',
           'اعلا',
           'منسوبین',
           'قول',
           'اردو',
           'ده',
           'کیلوگرا',
           'مواد',
           'منفجره',
           'ده',
           'جنگ',
           'افزار',
           'سبک',
           'سنگین',
           'سو',
           'جنگجو',
           'گر',
           'وه',
           'طالب',
           'یک',
```

بخش سوم:

در این بخش به پیاده سازی heapو zeap میپردازیم اما در ابتدا تابعی نوشته شده تحت عنوان create_posting_list

```
#Creating posting list using Dictioanry


def creatte_postngs_list(data):
    frequency={}
    diction={}

    for i in range(len(data)) :

        for words in data[i]:

            if (words not in frequency):
                frequency[words] = 0
                frequency[words] = frequency[words] + 1

            else:
                frequency[words] = frequency[words] + 1

            if words not in diction.keys():
                    diction[words] = set()
                    diction[words].add(i)
            else:
                diction[words].add(i)
    return frequency, diction


frequency5,diction_sample5=creatte_postngs_list(dat_forposting_sample5)
```

که طبق آنچه در صورت پروژه آمده بود به صورت dic ذخیره شده است تا سرعت بالا باشد در این جا پستینگ لیست مورد نظر ساخته میشود به این صورت که اگر ورد در آن نباشد آن ترم اضافه گشته و اگر موجود باشد شماره ترم آن ذخیره گشته است .

برای محاسبه freq کد زیر مورد استفاده قزاز گرفت

```
#Zipf law
import seaborn as sns
import matplotlib.pyplot as plt
def frequency(diction):
    listoffrequency=[]
        #print value
    lengths = [len(v) for v in diction.values()]
    return lengths


list_frequency_first_sample=frequency(diction_sample5)
```

که در آن برای هر کلمه تعداد تکرار آن محاسبه شده است

برای مثال برای لیست 15000 تایی و 5000 تایی در کل به تعداد زیر توکن موجود میباشد

```
]: print(sum(list_frequency_first_sample),sum(list_frequency_secound_sample))
634906 1894815
```

```
04]: term_freq_df
ut[104]: [41808,
 22139,
 2073,
 10089,
 352,
 12363,
 536,
 971,
 782,
 3,
 4933,
 1118,
 913,
 13,
 13952,
 5,
 6165,
 2,
 2621,
```

برای کل دیتا ست

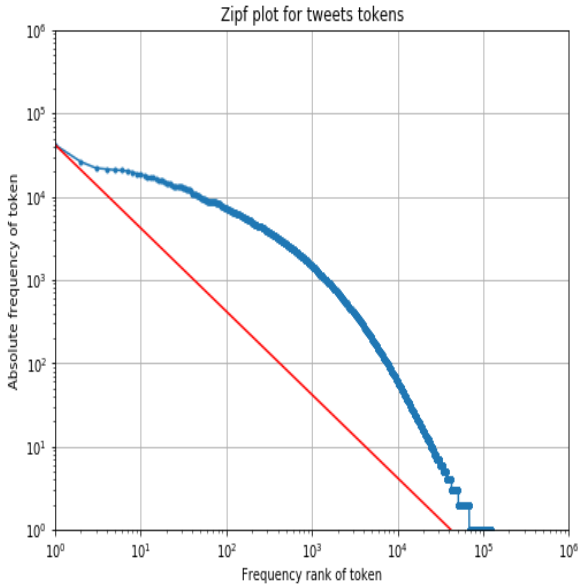
در کل برای هر دو قانون حالت دوم بهتر از حالت اول است بخاطر اینکه بهتر نرملایز شده و بهتر tokenizeانجام شده و همچنین ریشه یابی شده است این باعث میشود که تعداد توکن های منحصر به فرد کمتر شود که بسیار برای ما بهتر است و میتوان دید که هر دو قانون درست میباشند

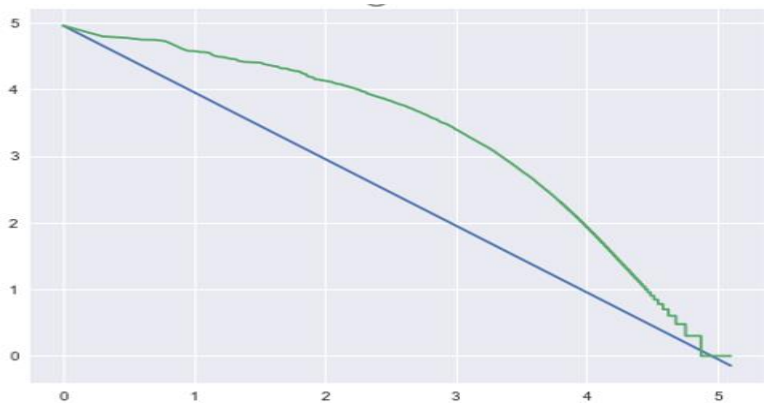
قانون Zipfیک قانون در مورد توزیع فراوانی کلمات در زبان است)یا در مجموعه‌ای که به اندازه کافی بزرگ است به طوری که نماینده زبان باشد .(برای نشان دادن قانون Zipfاجازه دهید فرض کنیم که ما یک مجموعه داریم و اجازه می‌دهیم در مجموعه کلمات منحصر به فرد)دایره لغات(باشد.

برای پیاده سازی آن همانطور که در کد نیز مشاهده میشود شکل زیر بدست آمد

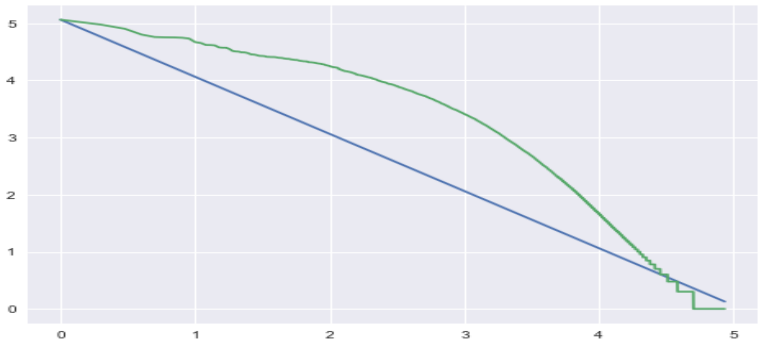
نمودار ها با seasbornکشیده شده اند

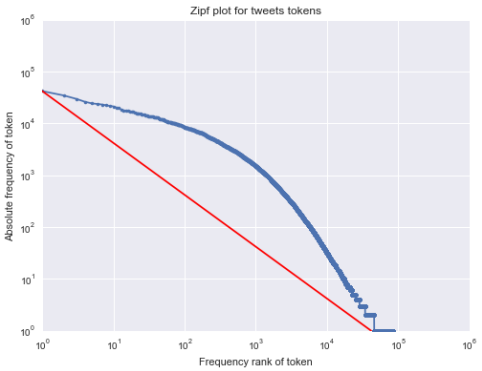
نمودا ر حالت 1





نمودار در حالت دوم





در این بخش نیز heapپیاده سازی شده است

```
import math as m
def Heap_law(list_sample_5,list_sample_10):
    print(m.log(list_sample_5/list_sample_10,10))
    print(m.log(1/3,10)/m.log(list_sample_5/list_sample_10,10))

    b=(m.log(1/3,10))/(m.log(list_sample_5/list_sample_10),10)
    print(b)

    t =15000/(pow(sumof,b))
    return b,t


b,k=Heap_law(sum(list(frequency5.values())),sum(list(frequency15.values())))
print(b,k)
```

که در اینجا دو زیرمجموعه 5000 تایی و 15000 تایی به صورت رندوم با تابع

```
import random
def random_sample(x,data):
    p=random.sample(range(len(data)), x)
    listofsample=[]
    for i in range(len(p)):
           print(p[i])
           listofsample.append(data[p[i]])
    return listofsample


list_sample_5=random_sample(5000,data['content'])
print(" first random sample done ")
list_sample_10=random_sample(15000,data['content'])
print(" secound random sample done ")
dat_forposting_sample5=pd.Series(list_sample_5)
dat_forposting_sample10=pd.Series(list_sample_10)
```

```
34694
```

ساخته شده که با توجه به عدد رندوم انتخاب میشود که چه داکیومنت هایی برای posting list انتخاب شوند

```
import math as m
def Heap_law(list_sample_5,list_sample_10):
    print(m.log(list_sample_5/list_sample_10,10))
    print(m.log(1/3,10)/m.log(list_sample_5/list_sample_10,10))

    b=(m.log(1/3,10))/(m.log(list_sample_5/list_sample_10),10)
    print(b)

    t =15000/(pow(sumof,b))
    return b,t


b,k=Heap_law(sum(list_frequency_first_sample()),sum(list_frequency_secound_sample))
print(b,k)
```

در اینجا نیز مقدار بهینه b و k با توجه به دو معادله دو مجهول بدست می‌آید

باسمه تعالی

مالی‌شرکتی

نام و نام خانوادگی: سپهر عسگریان **رشته تحصیلی: مهندسی کامپیوتر** **دانشگاه صنعتی امیرکبیر**