وزارت علوم، تحقیقات و فناوری
دانشگاه تحصیلات تکمیلی علوم پایه
گاوازنگ، زنجان

پروژه‌ی درس پردازش زبان طبیعی

سارینا دانایی
۹۸۴۱۳۱

استاد: دکتر ابراهیم انصاری

پاییز ۹۸

تعریف مسئله:

در این پروژه قصد داریم که جست‌وجورا میان ۱۴۰۰ داکیومنت را انجام دهیم . به صورتی که برای هر query که در یک فایل جداگانه موجود میباشد . بهترین جواب‌ها که در اینجا به معنای مرتبط ترین داکیومنت ها می‌باشد را برگردانیم.

که باید این کار را با دو روش که یک روش معمول و یک روش گفته شده در مقاله ی salton & backley میباشد انجام دهیم و خروجی را در نهایت مقایسه کنیم.

توضیح کد:

در **بخش ـ۱** فایل cran که شامل ۱۴۰۰ داکیومنت می‌باشد را باز میکنیم و mode را برابر r قرار می‌دهیم که به معنای read یا خواندن می‌باشد و در این فایل نمیتوانیم عمل نوشتن انجام دهیم.

سپس فایل را در یک متغیر به نام token قرار می‌دهیم که یک لیست می‌باشد و هر خانه‌ی جدول شامل یک کلمه ی داکیومنت می‌باشد. (با دستور split کلمات داکیومنت را براساس space از هم جدا کردیم و در لیست قرار دادیم)

در **بخش ـ ۲** با استفاده از تابع FreqDist که یکی از توابع کتابخانه nltk می‌باشد تعداد هر کلمه در لیست token را چاپ می‌کنیم (با استفاده از دستور freq.most_common() به صورت نزولی چاپ می‌شوند)

در **بخش ۳** با دستور porterStemmer تمامی کلمات stem می‌شوند به این معنا که تمامی پسوند های کلمات مثل ...,es,ed,ing از کلمات حذف میشوند.

نکته ۱: در این دستور از یک حلقه استفاده شده است که ۵ بار متوالی کلمات stem شوند. به این دلیل که در یک بار استفاده از این دستور یک سری از کلمات stem نشدند و ۵ بار دستور را اجرا کردم که جواب بهتری در نهایت به دست آمد اما زمان اجرا افزایش یافت.

در **بخش ۴** لیست stopwordها چاپ شده که شامل ۱۷۹ کلمه می‌باشد.

سپس تمامی stopwordها با استفاده از دستور removeاز داکیومنت ها حذف شدند. که در این دستور نیز به همان دلیل نکته ۱ پنج بار دستور remove انجام شده.

در متغیر tag یک لیست تعریف شده که تگ هایی که در داکیومنت هستند را در لیست وارد کردیم. این تگ ها نیز با استفاده از دستور remove حذف شدند.

در **بخش ۵** میخواهیم که این ۱۴۰۰ داکیومنت را از هم جدا کنیم و دوباره در یک لیست میگذاریم به صورتی که بتوانیم با استفاده از ایندکس که همان شماره داکیومنت میباشد .داکیومنت ها را به صورت جداگانه بخوانیم.در این قسمت split را براساس I. انجام میدهیم چون داکیومنت ها براساس I. از هم جدا شده اند.

در **بخش ۶** با استفاده از pandas یک DataFrame تعریف کردیم با نام tf که ستون های آن از ۱ تا ۱۴۰۰ (شماره ی داکیومنت ها) و سطرهای أن شامل type ها(کل کلمات داخل داکیومنت میباشد)

سپس با استفاده از حلقه تعداد تکرار هر کلمه را در هر داکیومنت به دست آوردیم و برطول لیست شامل داکیومنت ها تقسیم کردیم تا مقدار tf را به دست أوریم.

سپس دوباره یک DataFrame دیگر با نام idf میسازیم که شامل ۲ ستون میباشد :

ستون اول باید مقدار idf درآن قرار گیرد با نام idf

و ستون دوم که با نام counter مشخص شده است تعداد تکرار هر کلمه که در سطر ها قرار دارد در کل ۱۴۰۰ داکیومنت نشان میدهد (یعنی هر کلمه در کل داکیومنت چند بار تکرار شده است)

برای مقدار idf هر کلمه باید تعداد داکیومنت ها را بر تعداد تکرار آن کلمه در کل داکیومنت هاکه همان ستون counter میباشد. به دست آوریم وسپس لگاریتم میگیریم.

در **بخش ۷** دقیقا مانند بخش ۶ عمل کرده(DataFrameبا اسم best_pro) با این تفاوت که در اینجا با توجه به مقاله ی salton and buckley مقدار tf و idf را از یک فرمول دیگر به دست می آوریم به این صورت که برای tf از فرمول :

$$0.5 + (0.5 * \frac{tf}{\max(tf)})$$

استفاده میکنیم

و برای idf از فرمول:

$$log10(N - \frac{n}{n})$$

استفاده میکنیم.

در این قسمت سوال اول assignment 2 تمام میشود.

در **بخش ۸** که مربوط به سوال دوم می باشد فایل queryها را مانند فایل cran باز میکنیم.

در **بخش ۹** با استفاده از regular exoression اعداد که همان شماره ي داکیومنت ها میباشند و تگ W. را از لیست query ها حذف میکنیم.

سپس همانند داکیومت ها با استفاده از تگ I. لیست queryها را split میکنیم تا به هر query به صورت جداگانه به وسیله ی index دسترسی داشته باشیم. در ادامه تا رسیدن به بخش

۱۰. کارهایی مانند حذف stopwprd ها و stemming انجام شده است. (توضیحات در قسمت کامنت ها)

در **بخش ۱۰.** یک DataFrame با نام j_query ساخته شد که ستون ها . شماره ی query ها میباشد و سطرها شماره ی داکیومنت ها.

در **بخش ۱۱.** برای هر خانه ی جدول که یک query را نشان میدهد برای هر query تک تک کلماتش جدا میشوند و مقدار idf*tf میشود این مقدار را برای تک تک کلمات هر queryبه دست می آوریم و در أخر با هم جمع میکنیم.

یعنی برای هر query مقدار tf*idf کل کلماتش را در هر داکیومنت حساب میکنیم و با هم جمع میکنیم

مقدار tfو idf در بخش ۶. محاسبه شده است.

در **بخش ۱۲.** نیز مانند بخش ۱۱. یک DataFrame با نام j_query_best ساخته شده با این تفاوت که برای مقدار tf*idf هر خانه از مقادیر tfو idf بخش ۷. استفاده کردیم.

در **بخش ۱۳.** فایل relevant باز و چاپ شده است.

این لیست شامل شماره ی داکیومنت هایی است که برای تعداد ۱۰ query که در سوال خواسته شده. (این queryها شماره ی ۱۰ تا ۱۹ لیست queryها میباشند)

در **بخش ۱۴.** میخواهیم مقدار precision و recall را برای هر query با هر دو روش به دست آوریم (روش معمول و یکی از روش های گفته شده در مقاله ی salton &backley)

برای این کارابتدا برای هر queryاز ۱۰ query جدولی که مجموع مقدار tf*idf را حساب کردیم(j_ query) به صورت نزولی sort میکنیم و با relevant مقایسه میکنیم و شماره هایی از داکیومنت ها که در لیست sort شده آمده و در relevant موجود میباشد را حساب میکنیم و precisionو recallرا محاسبه و چاپ کرده. (این کار ابتدا برای ۱۰ داکیومنت اول سپس برای ۵۰ و ۱۰۰ و ۵۰۰ داکیومنت sortشده ی اول انجام شد)

این کار را برای روش دوم نیز انجام داده و precisin و recall را محاسبه میکنیم.

با مقایسه ی precison در این دو روش مشاهده میشود که روش گفته شده در مقاله که در این کد با نام j_query_best تعریف شده است بهتر عمل میکند.

بهتر عمل کرده (یعنی تعداد بیشتری از داکیومنت های لیست relrvantرا برای هر query برگردانده است).