**به‌نام خدا**

1. **کلاس Index**

این کلاس وظیفه ساخت ایندکس، بازیابی اسناد مرتبط و رتبه‌بندی اسناد را انجام می‌دهد. در ادامه توابع اضلی این کلاس شرح داده می‌شود:

* تابع parse:

این تابع تمامی اسناد را پردازش نموده و دیکشنری و posting\_list مربوط به مجموعه داده را ایجاد می‌کند. به‌منظور پردازش متن فارسی از کتابخانه‌ی پارسی‌وار[[1]](#footnote-1) استفاده شده است. پارسی‌وار یک جعبه‌ابزار پیش‌پردازش زبان فارسی است که برخی از وظایف مهم مانند نرمال‌سازی، ریشه‌یابی، بن‌واژه‌سازی[[2]](#footnote-2) و مانند آن را انجام می‌دهد. کار با این ابزار مشابه ابزار هضم است اما از دقت بالاتری نسبت به آن برخوردار است. به‌منظور آماده‌سازی محتوای فایل‌های متنی برای پردازش، ابتدا علائم ویژه و اعداد با استفاده از عبارات منظم از متن حذف شده‌اند. سپس کلمات توقف[[3]](#footnote-3) که فاقد بار معنایی متمایزکننده هستند، از متن حذف می‌شوند. برای انجام این کار از یک فایل آماده که حاوی فهرستی از کلمات توقف فارسی است استفاده شده است[[4]](#footnote-4). در ادامه لغات و نام اسنادی که در آن رخ داده‌اند به یک شناسه عددی یکتا تبدیل می‌شود. برای این کار از یک کلاس کمکی به‌نام IDMap استفاده شده است. توابع این کلاس عملیات نگاشت نام به شناسه و برعکس را انجام می‌دهند. با کمک این کلاس، به‌ازای هر لغت موجود در متن یک تاپل به‌صورت (term\_id, doc\_id) ایجاد می‌شود.

برای نگهداری کلمات دیکشنری از ساختار B-Tree استفاده شده است. به همین منظور کتابخانه OOBTree در پایتون مورد استفاده قرار گرفته است. لغات موجود در پیکره به‌صورت کلید در این ساختار ذخیره می‌شوند و posting\_list مربوط به هر لغت به‌عنوان مقدار آن کلید ذخیره می‌شود. به‌ازای هر یک از تاپل‌های ایجاد شده در متن بررسی می‌شود که آیا آن term در درخت قرار دارد یا خیر. در صورتی که کلمه موجود نباشد، یک کلید جدید به درخت اضافه می‌شود و یک posting\_list برای آن ایجاد می‌شود. اما اگر کلمه از قبل وجود داشته باشد، تنها شماره سند مربوط به آن در posting\_list متناظر با کلید کلمه اضافه می‌شود. با توجه به این‌که برای نگهداری posting\_list از ساختار set() استفاده شده است، بنابراین عناصر تکراری به‌صورت خودکار حذف می‌شوند. زمانی‌که تمام متن‌های موجود در پیکره پردازش شود، ساخت ایندکس به پایان می‌رسد.

* تابع retrieve:

این تابع با توجه به کوئری دریافت‌شده از کاربر، اسناد مرتبط را جمع‌آوری می‌کند. به همین منظور لازم است در ابتدا کوئری با استفاده ابزار پارسی‌وار پیش‌پردازش شود. پس از پردازش کوئری اسنادی که حاوی حداقل یکی از کلمات پرسش هستند جمع‌آوری شده و به ماژول رتبه‌بندی ارسال می‌شوند. با این کار اسناد نامرتبط حذف شده و فضای جستجو محدودتر می‌شود که موجب افزایش سرعت اجرا خواهد بود.

* تابع ranker:

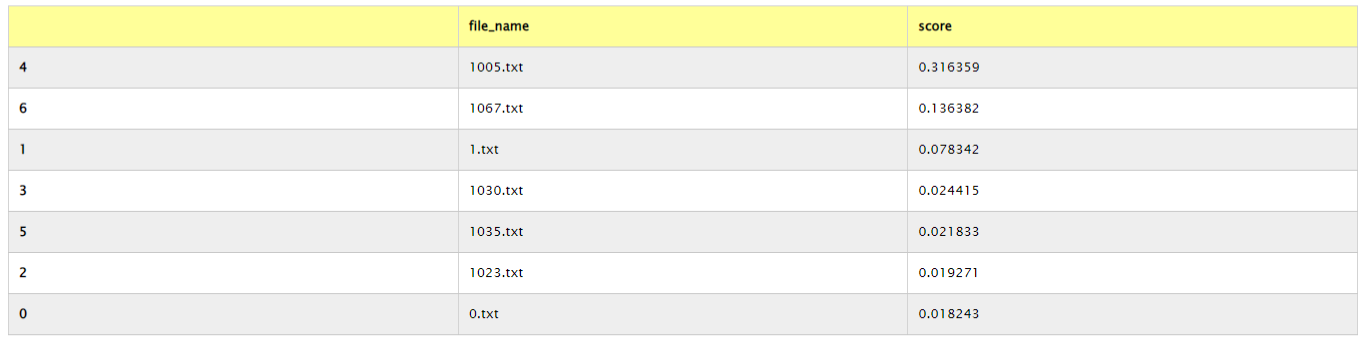
این تابع مجموعه اسناد دریافت‌شده از تابع retrieve را رتبه‌بندی می‌کند. به همین منظور پرسش کاربر و محتوای اسناد به بردارهای TF-IDF تبدل می‌شوند. برای تبدیل متن به بردار از ماژول TfIdfVectorizer از کتابخانه‌ی scikit learn استفاده شده است. پس از تبدیل متن به بردار، با استفاده از ماژول cosine\_similarity از این کتابخانه، مشابهت پرسش با هر سند اندازه‌گیری می‌شود. در پایان مجموعه اسناد مرتبط به‌صورت یک دیتافریم مرتب‌شده بر اساس امتیاز مشابهت بازگردانده می‌شود.

1. **مشاهده نتایج**

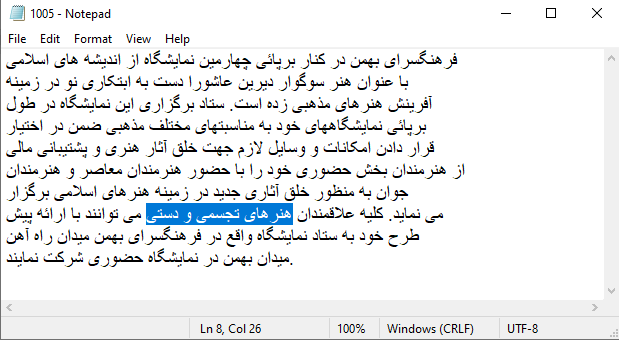
به‌منظور مشاهده نتایج به‌صورت گرافیکی از چارچوب فلسک استفاده شده است. با توجه به شکل زیر کاربر می‌تواند کوئری مورد نظر خود را تایپ کرده و سپس کلید جستجو را انتخاب کند. برای مثال در این شکل عبارت «هنرهای تجسمی و دستی» وارد شده است.



در نتیجه‌ی کوئری مذکور، فهرستی حاوی شناسه و نام فایل اسناد مرتبط نمایش داده می‌شود که بر اساس میزان شباهت کسینوسی مرتب شده است:



با توجه به فهرست بازیابی‌شده در شکل فوق، فایل 1005.txt بیشترین میزان مشابهت با پرسش را داشته است. با توجه به محتوای این فایل، متن سند حاوی کلمات موجود در پرسش ورودی بوده است.



1. Parsivar [↑](#footnote-ref-1)
2. Stemming [↑](#footnote-ref-2)
3. Stopwords [↑](#footnote-ref-3)
4. <https://github.com/kharazi/persian-stopwords> [↑](#footnote-ref-4)