گزارش فاز سوم پروژه بازیابی پیشرفته اطلاعات- دکتر بیگی

شبنم قاسمی راد ۹۴۱۰۵۸۰۳

پرند علیزاده علمداری ۹۴۱۰۰۰۲۴

تمام قسمت‌ها مطابق مطالب درس پیاده سازی شده است. تقسیم کار در این پروژه به صورت مساوی انجام شده است. زبان برنامه نویسی پایتون است.

در ادامه توضیح بخش‌های مختلف پروژه آمده است.

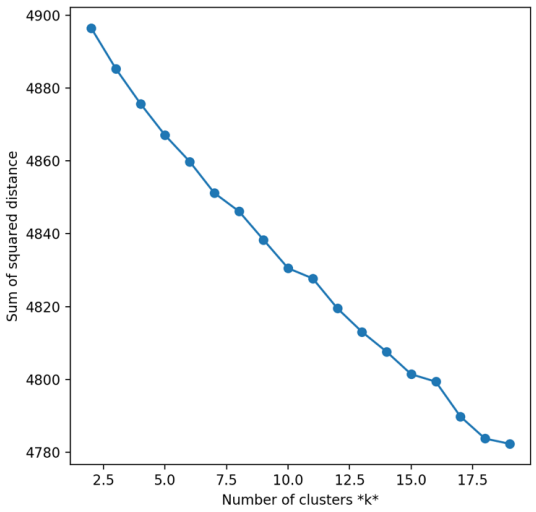
خوشه بندی

این قسمت با استفاده از توابع موجود در sklearn و genism پیاده سازی شده است. همچنین برای word2vec از لغات آموزش یافته google استفاده شده است.

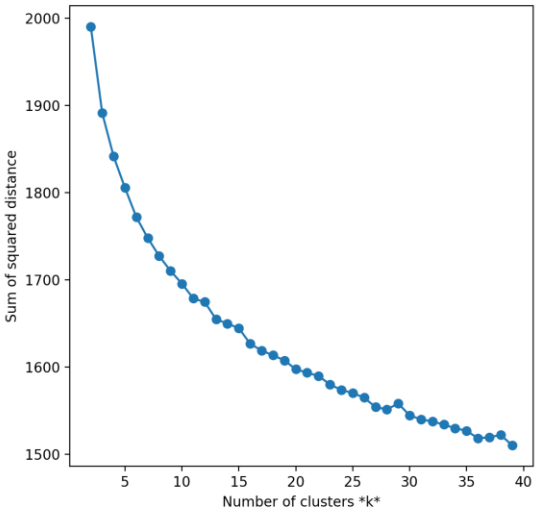
Kmeans:

برای این قسمت برای هر دو نوع بردار k های مختلف امتحان شد و با توجه به مجموع فواصل نقاط و قانون elbow، مقدار k انتخاب شد. برای هر دو بردار این مقدار برابر با ۱۰ انتخاب شد. نمودار هر دوی این‌ها در ادامه آمده است.

نمودار tf-idf



نمودار word2vec

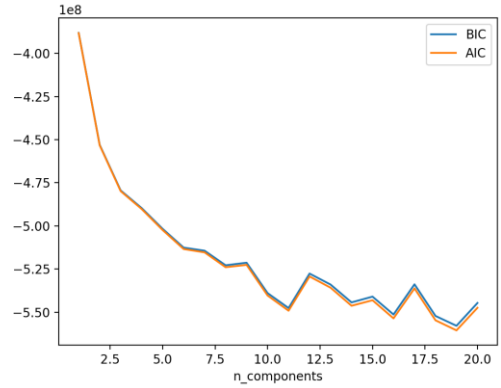


GMM:

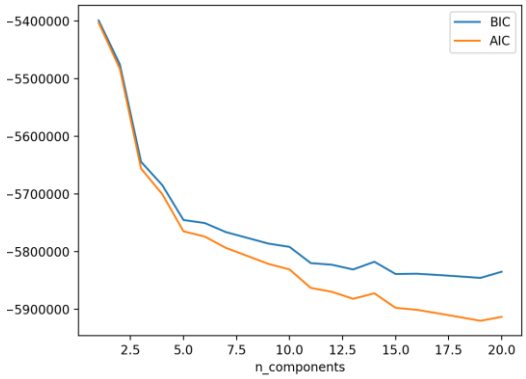
برای هر دو بردار و تعداد کلاسترهای مختلف (بین ۱ تا ۲۱) اجرا شد و covariance بعد از تعدادی تست diag ست شد. با استفاده از تحلیل aic و bic تعداد کلاستر مناسب انتخاب شد.

برای tf-idf این مقدار برابر با ۱۱ و برای word2vec برابر با ۱۵ انتخاب شد.

نمودار tf-idf



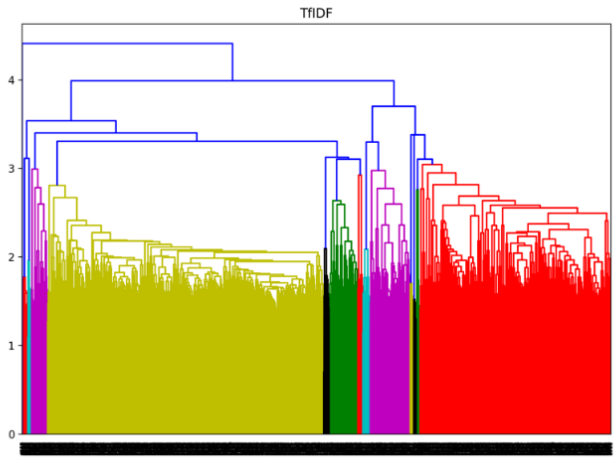
نمودار word2vec



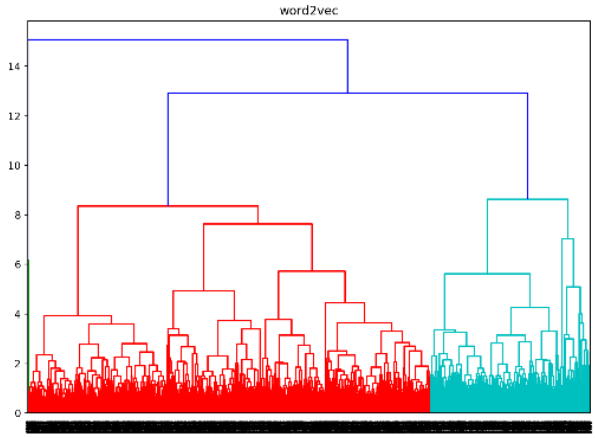
Hierarchical:

در این قسمت هم ابتدا دندروگراف رسم شد و تعداد کلاستر با استفاده از آن انتخاب شد. و درنهایت با استفاده از Aggolmarative Clustering خوشه بندی شدند.

نمودار دندروگراف tf-idf



نمودار word2vec



خزش

در این بخش با استفاده از کتابخانه‌ی scrapy یک خزشگر پیاده سازی کردیم. ابتدا با دستور scrapy startproject و وارد کردن crawler به عنوان نام پروژه‌ی خزشگرمان، به صورت اتوماتیک فولدرهای spider و فایل‌های تنظیمات ساخته شد و تاخیر فرستادن درخواست را نیز در فایل settings.py به عدد0.25 ست کردیم. در غیر این صورت به دلیل نزدیکی زیاد درخواست‌ها با پیغام ۴۰۳ مواجه می‌شدیم. در فایل crawler.py بدنه‌ی اصلی خزشگر نوشته شده است.

خزشگر با دستور scrapy crawl semantic_scholar ران می‌شود و برای ذخیره‌ی خروجی در فایل جیسون کافی است این دستور را به scrapy crawl semantic_scholar –o crawled_papers_info.json تغییر دهیم. اما برای اینکه لزوما نیازی به این کار نباشد، کار ذخیره در فایل جیسون را در کد به صورت دستی نیز قرار دادیم.

در تابع start_requests آدرس صفحاتی که در فایل start.txt قرار داشتند در start_ids قرار گرفته است و به عنوان دانه های شروع از آنها استفاده می‌شود.

در تابع parse، با استفاده از css selector ها ارجاعات هر مقاله بازیابی شده است که ۵ مورد اول با استفاده از دستور response.follow در صف خزش قرار می‌گیرند. اطلاعات خواسته شده نیز با استفاده از همین سلکتورها بدست آمده‌اند. در مورد چکیده و نویسندگان برای اینکه نیازی به زدن دکمه show_more نباشد، از تگ meta شامل چکیده‌ی کامل و لیست کل نویسندگان است استفاده شده است.

رتبه بندی

در فایل pagerank.py تابع get_page_rank با مقدار دیفالت آلفا = ۰,۱۵ وجود دارد که می‌توان آلفای دیگری را نیز به آن به عنوان پارامتر ورودی داد. ابتدا مقالات واکشی شده از فایل جیسون لود شده است و سپس گرافی را با توجه به reference های مقاله ساخته‌ایم. سپس ماتریس P را ، که ماتریس adjacency ای برای گراف است که هر سطر با تعداد ناصفر ۱، با توجه به تعداد ۱ ها نورمالایز شده است و سطرهای تماما صفر با تماما 1/n جایگزین شده است، ساختیم و با استفاده از این ماتریس، با توجه به فرمول $\mathbf{P} = 1 - \alpha\,\mathbf{P} + \alpha\mathbf{v}$ ، آن را نورمالایز کردیم و سپس به روش iterative هر بار (حداکثر تا ۵۰۰ بار) با شروع از ماتریس احتمالات $a = (1, 0, 0, \ldots, 0)$ و هر بار ضرب کردن آن در ماتریس P، زمانی که اختلاف دو ایتریشن از 10^{-9} کمتر شد، a را به عنوان خروجی نهایی رتبه بندی چاپ می‌کنیم، زیرا همگرا شده است. نتایج این رتبه بندی همچنین در فایل page_ranks.txt ذخیره شده است.

توضیحات تکمیلی

فایل‌های csv مربوط به بخش اول در فولدر clustering-output قرار داده شده است.