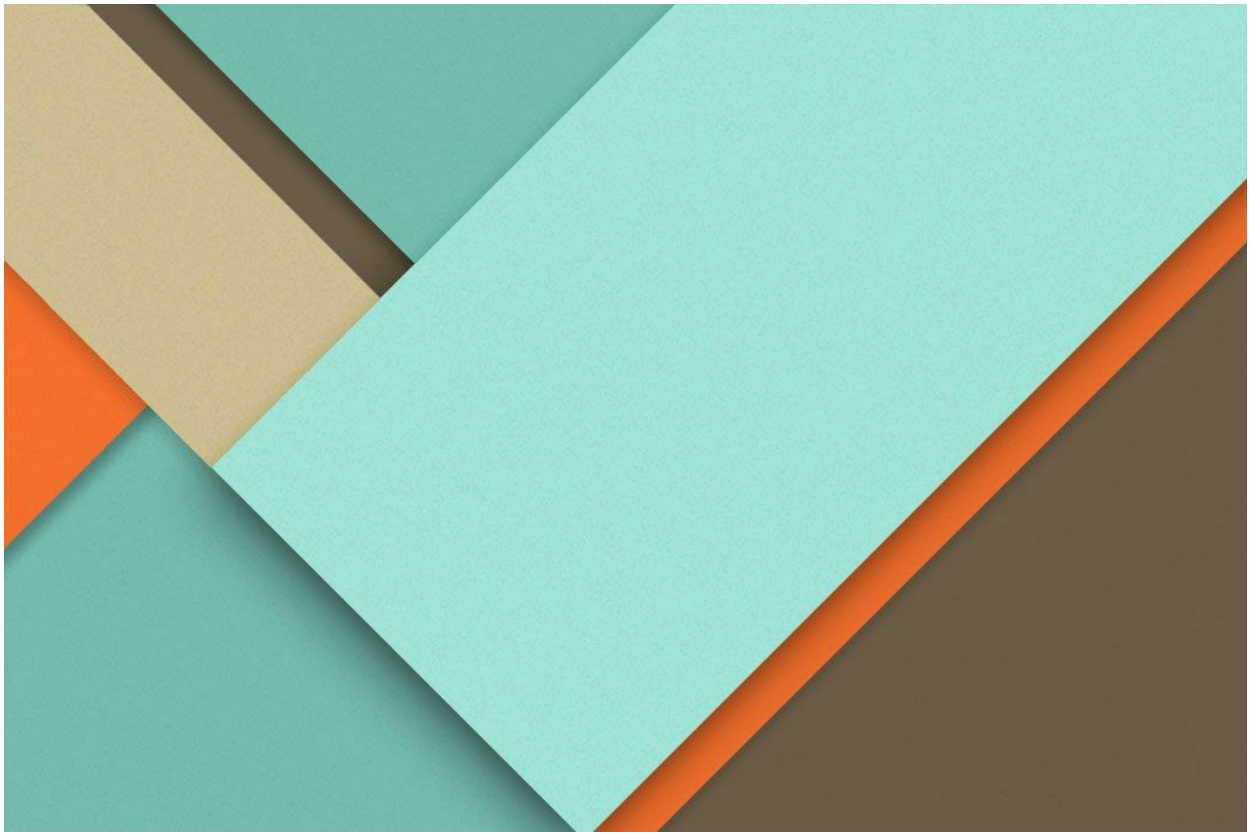


# بازیابی اخبار فارسی

مرداد ۹۹

محمدمهدی رحیمی
۹۳۲۳۰۹۳

## مقدمه

در این پروژه یه سیستم بازیابی اطلاعات برای داده های فارسی یک سایت خبری طراحی شده است. این سیستم بعد از مراحل پیش پردازش داده ها مبتنی بر ریشه کلمات و امتیاز tf-idf آنها را بازیابی می کند. در این گزارش به نحوه پیاده سازی و عملکرد و همچنین تشریح قسمت مختلف سیستم بازیابی اطلاعات می پردازیم.

## ساختار پروژه

این پروژه با زبان پایتون ۳ و با ابزار jupyter-notebook توسعه یافته و برای اجرا و تغییر آن این روش توصیه می‌شود. همچنین کد پایتون و قالب html آن نیز خروجی گرفته شده است تا بدون این ابزار نیز بتوان پروژه را اجرا و یا بررسی کرد.

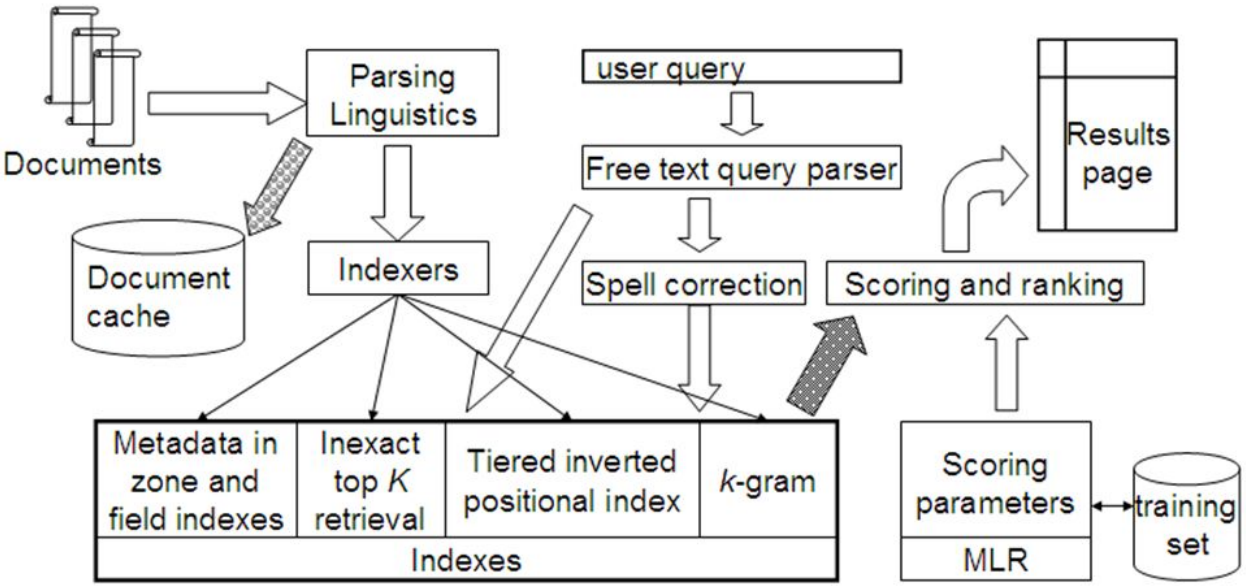
کتابخانه های مورد استفاده در این پروژه به شرح زیر می‌باشد:

| ردیف | نام کتابخانه | توضیحات |
|---|---|---|
| ۱ | BeatifulSoup | حذف تگ های html و کد های جاوا اسکریپت از متن |
| ۲ | Pandas | بارگذاری فایل های و اعمال تغییرات بر روی داده ها |
| ۳ | Re | پیدا کردن الگو ها در متن برای حذف و یا جایگزینی |
| ۴ | Matplotlib | کشیدن نمودار ها |
| ۵ | Random | ساخت اعداد تصادفی برای نمونه برداری |
| ۶ | Functools | کمک به کار با لیست های تو در تو |
| ۷ | Operator | کمک به یک سطحی کردن لیست های تو در تو |
| ۸ | Numpy | توابع ریاضی لگاریتمی و محاسبه مقدار تجمعی لیست ها |
| ۹ | OS | دریافت لیست فایل های موجود برای پردازش |
| ۱۰ | PPrint | نمایش زیباتر خروجی های متنی |
| ۱۱ | Hazm | فقط برای ساختن توکن ها |
| ۱۲ | Parsivar | فقط برای ریشه‌یابی افعال |
| ۱۳ | PersianStemmer | فقط برای ریشه یابی کلمات |

فایل های موجود در این پروژه به شرح زیر می‌باشد:

| ردیف | نام فایل یا پوشه | توضیحات |
|---|---|---|
| ۱ | IR-S19-project-data/ | پوشه شامل محل قرار گیری فایل های برای پردازش |
| ۲ | IR.ipynb | دفترچه جوپیتر برای اجرای برنامه |
| ۳ | stopwords.txt | لیست کلمات پرتکرار کوتاه |
| ۴ | all.txt | لیست تمامی کلمات پرتکرار |
| ۵ | IR.html | صفحه html برای نمایش دفترچه و خروجی ها |
| ۶ | IR.py | کد نهایی پایتون |
| ۷ | expressions.txt | مجموعه ای از عبارت فارسی برای توکن سازی به طور صحیح تر |

## پیاده سازی




شکل-1 قسمت های سیستم بازیابی اطلاعات و نحوه ارتباط آنها

در این بخش به معرفی و نحوه پیاده سازی و عملکرد هر یک از این بخش ها می پردازیم.

1. Parsing Linguistics فاز اول : در بخش های واکشی خبر،استخراج توکن، ریشه یابی، نرمال سازی و حذف کلمات پر تکرار صورت گرفت
2. Free text query parser: فاز اول: مراحل انجام شده برای مستندات بر روی پرسمان نیز اعمال شده است.
3. Spell Correction وجود ندارد.
4. Indexers فاز اول: ساهت لغتنامه و شاخص معکوس از کلمات به مستندات انجام شد / **فاز دوم: محاسبه مقدار مورد نیاز جهت رتبه بندی افزوده شد**
5. Scoring and Ranking **فاز دوم: روش tf-idf به کمک ساخت لیست فهرمان پیاده سازی شد**
6. Scoring Parameters **فاز دوم: به صورت ثابت در نظر گرفته شده است ( پیاده سازی خاصی وجود ندارد.)**
7. Document Cache وجود ندارد.

# فاز اول:

## واکشی خبر

به کمک کتابخانه Pandas فایل ها csv موجود خوانده شده و همگی در قالب یک DataFrame قرار گرفته است.

سپس به کمک کتابخانه BeautifulSoup تمامی تگ های html و اسکریپت های جاوا اسکریپت از متن اصلی حذف شده و متن به صورت رشته در آماده است.

کاراکتر های نیم‌فاصله به کاراکتر فاصله تبدیل شده و تمامی فاصله های بیشتر از یک فاصله یا خط جدید و کاراکتر های علایم و نشانه ها حذف شده اند.

## استخراج توکن ها

در **حالت اول** صرفا با جداسازی هر کلمه به کمک کاراکتر فاصله توکن ها استخراج شده است.

در **حالت دوم** به کمک کتابخانه Hazm توکن ها استخراج شده است و کلماتی مانند رفته است به توکن های «رفته»، «است» و «رفته_است» تغییر کرده و ذخیره شده است. این کتابخانه به کمک لیست از پیش تعریف شده و قوانین دستوری زبان فارسی افعال را به شکل صحیح به توکن ها تقسیم می‌کند اما برای عبارات زبان فارسی هیچ کمکی نمی‌کند. همچنین اگر کلمات فارسی به اعداد و حروف انگلیسی چسبیده باشند به طور صحیح توکن ها تشخیص داده نمی‌شود.

پس در یک مرحله قبل ابتدا کلمات فارسی را از اعداد و حروف انگلیسی جدا کرده و بعضی از عبارات زبان فارسی با خط زیر به هم پیوند داده می‌شود.

**سوال: آیا میتوانید روشی خودکار برای استخراج این نوع عبارات پیشنهاد کنید؟**

بله؛ در صورتی که یک لغتنامه از تمامی ترکیب های چندتایی کلمات استخراج کنیم. اما حجم لغت‌نامه برای استخراج تمامی ترکیب ها نسبت به تعداد کلمات در ترکیب به صورت نمایی بالا میرود و اینکار برای بیش از دو یا نهایتا سه کلمه ممکن نیست. اما میتواند استخراج ترکیب های کلمات را تنها حساس به کلمات پرتکرار در لغت‌نامه با ترکیب کوچیک تر بدست آورد تا حجم لغت نامه کاهش یابد.

## نرمال‌سازی

نرمال سازی به دو صورت حذف و تبدیل کاراکتر ها صورت گرفته است:

تمامی اعراب ها، ایموجی ها، نشانه ها، همزه های ناخوانا و حروف غیر از فارسی، عربی یا ریشه لاتین حذف شده است.

تمامی حروف فارسی با کاراکتر های گوناگون عربی و غیره به صورت یک نوع کاراکتر فارسی تبدیل شده است.

تمامی حروف انگلیسی و اعداد از هر نوع و زبان های نزدیک به آن به یک صورت حروف کوچک اسکی در آماده است.

در نهایت مجموعه کاراکتر ها با اعمال نرمال سازی از ۴۰۰ کاراکتر یافت شده (پس از تغییرات اولیه در واکشی خبر) به ۷۰ کاراکتر (۱۰ کاراکتر عددی، ۳۳ کاراکتر فارسی، ۲۶ کاراکتر انگلیسی کوچک و ۱ کاراکتر فاصله) کاهش یافته است.

## ریشه‌یابی کلمات

برای ریشه‌یابی کلمات از کتابخانه های Parsivar برای ریشه یابی افعال و PersianStemmer برای ریشه یابی لغات استفاده شده است. علت استفاده از کتابخانه های گوناگون در هر بخش دقت بالاتر آن کتابخانه برای آن‌کار بوده است.

کتابخانه Parsivar تمامی افعال را به صورت ماضی&مضارع تبدیل می‌کند و اینگونه ذخیره سازی نسبت به ذخیره تنها یک مورد ماضی یا مضارع فعل دارای ابهام کمتری است. به طور مثال اگر تنها حالت مضارع ذخیره سازی شود: فعل بساز و کلمه «ساز» در «ساز موسیقی» هر دو به یک لغت در لغت‌نامه تبدیل می‌شود. اگرچه در این روش باید کلمات پر تکرار برای حذف را نیز طبق این ساختار تعریف کنیم و یا از این کتابخانه برای ریشه‌یابی قبل از حذف استفاده کنیم.

کتابخانه PersianStemmer برای ریشه‌یابی کلمات استفاده شده است، علت استفاده از این کتابخانه در مقایسه با دیگر کتابخانه ها دقت بالا این کتابخانه است. برای مثال کتابخانه های hazm و Parsivar کلمات را به صورت ساده لوحانه و با حذف پسوند یا پیشوند ها پرتکرار زبان فارسی ریشه‌یابی میکنند و به طور نمونه همانطور که کلمه درختان به درخت تبدیل می‌شود کلمه کمان و کاشان به کم و کاش تبدیل میکنند، با اینکه این روش به شدت کمک در کم شدن حجم لغت‌نامه و شاخص میکند اما دقت پایین تری در جواب دادن به پرسمان ها را ایجاد می‌کند. کتابخانه PersianStemmer با جمع آوری یک پایگاه داده سعی در ریشه‌یابی با توجه به معنا لغات دارد.

| ردیف | ریشه | کلمات | ردیف | ریشه | کلمات |
|---|---|---|---|---|---|
| ۱ | گفت | گفت | ۹ | شنو | شنو |
| ۲ | گو | 'فراگوت','واگو','پیشگو','گو','گوان','گوانگ','گوت','گوترش','گوند','گوها','گوهای','گوهایش','گوهایمان','گوهایی','گوچی','گوین','گوینت' | ۱۰ | هنر | 'هنرهاست', 'هنرش', 'هنرانی', 'هنرست', 'هنرها', 'هنرواره', 'هنرشان', 'هنرهای', 'هنر', 'هنرهایی' |
| ۳ | رو | 'روباره', 'روست', 'روها', 'روهای', 'روین', 'روترهای', 'روت', 'روترها', 'روهاست', 'فرارو', 'روهایی', 'روتر', 'رو' | ۱۱ | دوست | 'دوست','دوستان','دوستانش','دوستانشان','دوستانم','دوستانمان','دوستانی','دوست ت','دوستش','دوستشان','دوستم' |
| ۴ | رود | 'رودهایی', 'رودهای', 'رود', 'رودی', 'رودها' | ۱۲ | توان | 'بازتوانی', 'توان', 'توانست', 'توانند', 'توانان', 'توانایی' |
| ۵ | خواه | 'خواه', 'خواهنده', 'خواگین', 'خواهانه', 'واخواهی', 'بازخواهی', 'بازخواهیم' | ۱۳ | شریف | 'شریفیان', 'شریفی', 'شریفین', 'شریفش', 'شریفترین', 'شریف', 'شریفشان' |
| ۶ | سپاس | 'سپاسی', 'سپاس' | ۱۴ | یاد | 'یاد','یادان','یادت','یادتان','یادش','یادشان','یادم','یادمان','یادها','یادی' |
| ۷ | کرد | 'کردیان', 'کردی', 'کردنده', 'کردیانی', 'کردهای', 'کردند', 'کردهام', 'کردت', 'بازکرد', 'کردمان', 'کردشان', 'کردها', 'کردش', 'کردین', 'کردهایی', 'کرد', 'کردهاست', 'کردم' | ۱۵ | ساز | 'سازهایی', 'سازانی', 'واسازی', 'سازترین', 'بازساز', 'سازان', 'سازی', 'سازم', 'سازانه', 'پیشساز', 'ناساز', 'سازهای', 'سازتر', 'سازهایشان', 'سازند', 'سازیهای', 'سازیان', 'ساز', 'سازانش', 'سازها', 'سازیها', 'سازهاست', 'سازست', 'سازواره' |
| ۸ | دان | 'دانی', 'دان', 'دانهای', 'دانهایی', 'دانانگ', 'دانالو', 'دانست', 'دانها', 'داننده', 'دانند', 'دانم', 'دانان' | | | |

جدول افعال تشخیص داده شده:

| ردیف | ریشه | کلمات |
|---|---|---|
| ۱۶ | کرد&کن | 'نکنیم', 'نمیکنند', 'میکنیم', 'میکنی', 'بکنم', 'نکرد', 'نمیکنیم', 'نکنند', 'کردیم', 'کنید', 'بکنیم', 'نمیکند', 'میکردم', 'بکنی', 'میکردی', 'نکردم', 'کنم', 'بکن', 'میکردیم', 'نکردیم', 'نکن', 'نکنید', 'کنیم', 'کنند', 'نمیکنم', 'کردید', 'نمیکردی', 'نکردهایم', 'خواهدکرد', 'نکردهاند', 'نمیکردند', 'میکنید', 'نکنم', 'نمیکردیم', 'نمیکنید', 'میکرد', 'بکنند', 'نکردی', 'میکنند', 'میکنم', 'بکنید', 'نکنی', 'میکردند', 'نکردند', 'کردهاند', 'نمیکرد', 'میکند', 'میکردید', 'خواهندکرد', 'نکردید' |
| | ساخت&ساز | 'بسازی', 'ساختیم', 'بساز', 'نساختم', 'بسازیم', 'سازد', 'نسازند', 'نسازی', 'میساخت', 'نساختی', 'میسازد', 'نسازید', 'بسازد', 'نسازد', 'بسازند', 'نسازم', 'سازیم', 'نساز', 'ساختهاند', 'سازید', 'بسازید', 'نسازیم', 'بسازم', 'ساختید' |
| | گفت&گو | 'بگویم', 'نگویم', 'نمیگویم', 'میگفت', 'نگفتید', 'میگفتم', 'گفتید', 'میگفتند', 'میگوید', 'نگفتیم', 'گفتی', 'نمیگوید', 'نگفتی', 'نگو', 'گفتند', 'نگفتند', 'گوید', 'نگوید', 'بگوید', 'بگوی', 'نگفت', 'گفتم', 'گفتیم', 'میگفتیم', 'بگو', 'نگفتم' |

## حذف کلمات پرتکرار

برای حذف کلمات پرتکرار لیست هایی از این کلمات تهیه شده است که شامل کلمات کوتاه، کلمات بلند، افعال، قیدها، صفات و عبارات فارسی است.

در حال حاضر یکبار فقط **کلمات پرتکرار کوتاه** و یک بار **تمامی کلمات پرتکرار** برای مقایسه حجم لغت‌نامه و شاخص معکوس حذف گردیده اند.

## ساخت لغت‌نامه

کلاس IndexMaker به طور همزمان لغت‌نامه و شاخص معکوس را می‌سازد. در ساخت لغت نامه تعداد تکرار هر لغت نیز ثبت می‌شود، که همچنین برای استفاده در بررسی قانون Zips مورد استفاده قرار می‌گیرد. همچنین اطلاعاتی در مورد تعداد کلمات دیده شده جدید و تعداد کل کلمات دیده شده برای بررسی قانون Heaps ذخیره می‌شود.

## ساخت شاخص معکوس

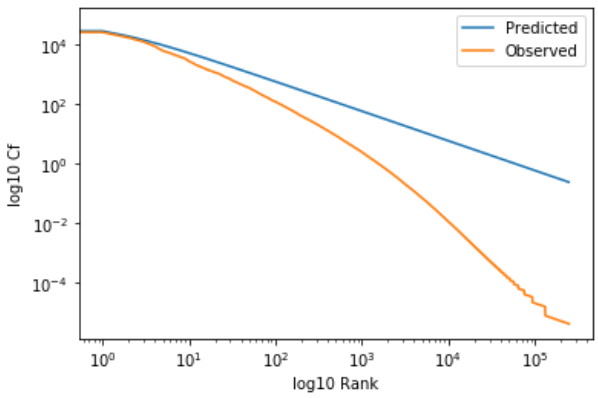
کلاس IndexMaker شاخص را با دریافت هر کلمه تکمیل میکند برای کاهش حجم شاخص از دیکشنری پایتون با هش کلمه به عنوان کلید استفاده شده است.

هر مقدار کلید مجموعه ای از مستنداتی که این کلمه آن کلید در آن یافت شده است را بازمی‌گرداند.

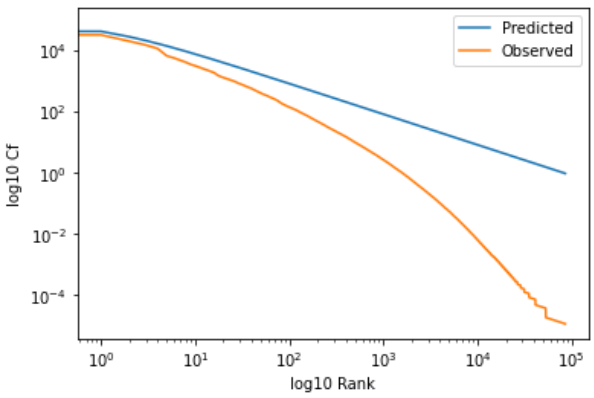
## محاسبه قانون Zipf

قانون Zipf ادعا دارد در هر زبان طبیعی نسبت تعداد تکرار کلمه به رتبه آن از نظر پرتکرار بودن نسبت معکوس دارد. ضریب این نسبت نیز تعداد تکرار پرتکرار ترین کلمه آن زبان است. از این رو تابع zipf که در کد نوشته شده است با دریافت لغت‌نامه استخراج شده است متن ها این موضوع را بررسی می‌کند.

حالت اول




حالت دوم



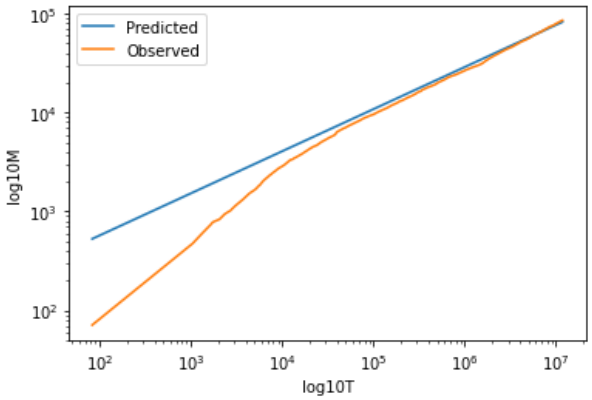

## محاسبه قانون Heaps

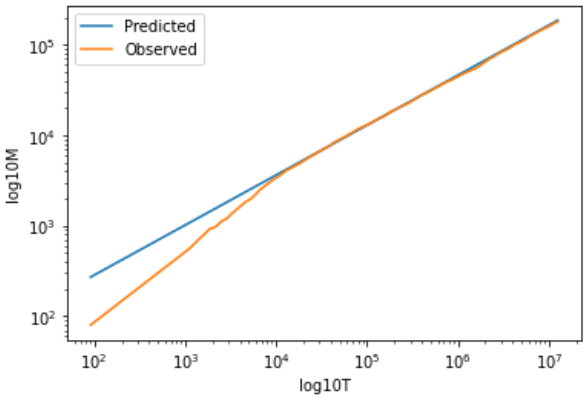
قانون Heaps ادعا میکند در هر زبان طبیعی رابطه بین تعداد توکن های استخراج شده با کلمات دیده شده به صورت نمایی است. بنابراین رابطه بین لگاریتم های آنها خطی است. برای بررسی این موضوع نیاز به دو بار نمونه برداری از داده ها برای محاسبه ضرایب داریم از این رو تابع heaps نوشته شده است تا با گرفتن لیست کلمات یک هر متن و تعداد توکن ها استخراج شده و کلمات جدید آن بتوانیم دو نمونه برداری از آن متن را داشته باشیم و نمودار های نهایی برای مقایسه پیش بینی قانون heaps با نتایج بدست آمده از متن خود را مقایسه کنیم.

نتایج در دو حالت بررسی شده به شرح زیر می‌باشد:

حالت اول

حالت دوم




## فاز دوم:

### محاسبه مقادیر tf و idf

لاس IndexMaker شاخص را با دریافت هر کلمه تکمیل میکند برای کاهش حجم شاخص از دیکشنری پایتون با هش کلمه به عنوان کلید استفاده شده است.

هر مقدار کلید مجموعه ای از مستنداتی که این کلمه آن کلید در آن یافت شده است را بازمی‌گرداند.

در فاز دوم این کلاس در حین ایجاد شاخص معکوس و لغت نامه مقدار idf (تعداد اسناد شامل کلمه) را محاسبه کرده و در لغت نامه قرار میدهد. همچنین یک ساختار دیکشنری پایتون دیگر برای ذخیر مقادیر tf (تعداد رخ داد هر کلمه در هر سند) به صورت مجزا تولید می شود.