

## تاثیر پردازش و مقایسه بین حالت اول و دوم:

در وهله اول می‌توان به تاثیر پردازش‌ها در حجم ذخیره‌سازی برای دیکشنری و posting list اشاره کرد که با انجام پردازش‌ها این حجم‌ها تا ۱۳٪ کاهش ایجاد کرد

این پردازش‌ها شامل حذف انواع کاراکترهای نیم‌فاصله که باعث یکسان‌تر شدن نوشتار واژه‌ها می‌شود و حذف اعداد و حذف حروف انگلیسی و حذف علائم نگارشی و حذف ایموجی‌ها هست که شاید خیلی در تعریف مسئله ما مورد اهمیت نباشد و باعث کاهش حجم و بهتر شدن نتایج برگردانده شده می‌شود و حذف انواع اعراب و یکسان‌سازی اشکال متفاوت نوشتاری حروف که باعث می‌شود تا برای برگرداندن نتیجه مجموعه‌ای از اسناد را به خاطر اشکال متفاوت نگارشی از دست ندهیم و بتوانیم نتیجه کامل‌تر و بهتری رو برگردونیم و ریشه‌یابی کلمات که باعث می‌شود افعال را فارغ از شناسه و ضمائر آن‌ها و کلمات را بدون توجه به جمع یا مفرد بودن نگاه کنیم تا بتوانیم نتایج بیشتر و بهتری داشته باشیم

حال به بررسی تعدادی مثال برای مقایسه بین حالت اول و دوم می‌پردازیم:

### ● جعبه چوبی

○ جعبه‌ی چوبی: خالی

○ جعبه چوب: 7498 , 11570 , 11812 , 31369 , 34196 , 41222 , 49955

### ● خبرگزاری‌ها گفتند

○ خبرگزاری‌ها گفتند: 7663 , 27906 , 32026 , 33726 , 35026

○ خبرگزاری: 0 , 2 , 7 , 8 , 13 , 14 , 16 , 24 , 27 , 28 , 29 , 34 , 36 ,

37 , 38 , 39 , 41 , 44 , 45 , ...

### ● خبرگزاری‌ها گفتند

○ خبرگزاری‌ها گفتند: خالی

○ خبرگزاری: 52870 , 53970

### ● خانه‌ی کلنگی

○ خانه‌ی کلنگی: خالی

○ خانه کلنگ: 971 , 1500 , 3711 , 4900 , 4963 , 9249 , 10168 ,

10463 , 11324 , 14587 , ...

### ● مسأله

○ مسأله: 85 , 464 , 790 , 892 , 1182 , 1252 , 1290 , 1534 , 2063 ,

2144 , 2283 , 2608 , ...

○ مساله: 62 , 69 , 85 , 94 , 111 , 188 , 233 , 464 , 564 , 666 , 740 ,

790 , 892 , 894 , ...

### ● می‌گذارد

○ می‌گذارد: 25 , 119 , 280 , 288 , 372 , 571 , 685 , 923 , 934 , 950 ,

985 , 1083 , 1135 , ...

○ گذار: 22 , 25 , 39 , 51 , 63 , 69 , 92 , 93 , 108 , 111 , 119 , 120 ,

121 , 129 , 142 , 152 , ...

## • می گذارد

○ گذارد: 180, 370, 668, 1145, 1210, 1466, 1666, 1827, 2265, 2336, 2372, 2381, ...  
○ گذار: 22, 25, 39, 51, 63, 69, 92, 93, 108, 111, 119, 120, 121, 129, 142, 152, ...

در مثال‌های ذکر شده می‌توان دید که پردازش‌ها باعث پیدا کردن اسناد بیشتری به ازای هر پرسمان شده است و تا حدی برای پرسمان‌های غیرمعمول مثل «خبرگذاری‌ها گفتند» بتوانیم نتایجی رو برگردانیم یا در مثال «مسأله» بتوانیم اسنادی که شامل اشکال نوشتاری متفاوتی از این کلمه هست را هم برگردانیم یا در مثال‌های «می‌گذارد» یا «می‌گذارد» اسنادی که شامل این فعل با شناسه یا ضمائر متفاوت هست را هم برگردانیم

## نرمال‌سازی:

- برای این قسمت تابع‌های زیر پیاده‌سازی کردم:
- `remove_half_spaces`: برای حذف نیم‌فاصله‌ها
  - `remove_numbers`: برای حذف ارقام
  - `remove_english_characters`: برای حذف حروف انگلیسی
  - `remove_punctuations`: برای حذف علائم نگارشی
  - `remove_diacritics`: برای حذف اعراب و تبدیل نوشتارهای متفاوت حروف
  - `remove_emojis`: برای حذف ایموجی‌ها
  - `replace_multi_words_token`: برای تبدیل عبارت‌های ترکیبی

## استخراج توکن‌ها:

برای این بخش در حالت اول بعد حذف کاراکترهای اضافی کلمات را بر اساس فاصله از هم جدا می‌کنم و توکن‌ها رو تولید می‌کنم و در حالت دوم از tokenizer کتابخانه `hazm` استفاده می‌کنم تا بهتر و با معنی‌تر توکن‌ها را استخراج کنیم

## استخراج خودکار عبارت‌های ترکیبی پرکاربرد:

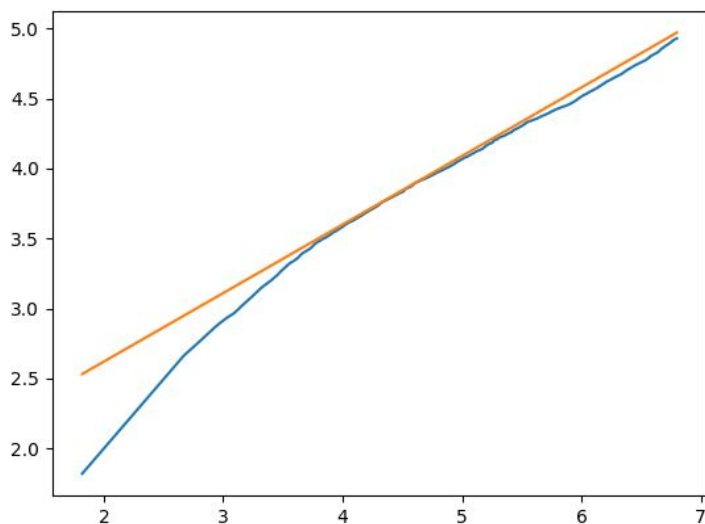
برای این کار می‌توان در حین ساختن شاخص همه زوج کلماتی که پشت هم ظاهر شده‌اند را بر اساس تعداد وقوع آن‌ها در تمامی اسناد از زیاد به کم مرتب کنیم و سپس با شروع از ابتدا و کنار گذاشتن زوج‌هایی که مثلاً حاصل از ترکیب یک حرف اضافه با یک کلمه دیگر هست به عبارت‌هایی ترکیبی برسیم و لیستی از آن‌ها درست کنیم

## ریشه‌یابی کلمات:

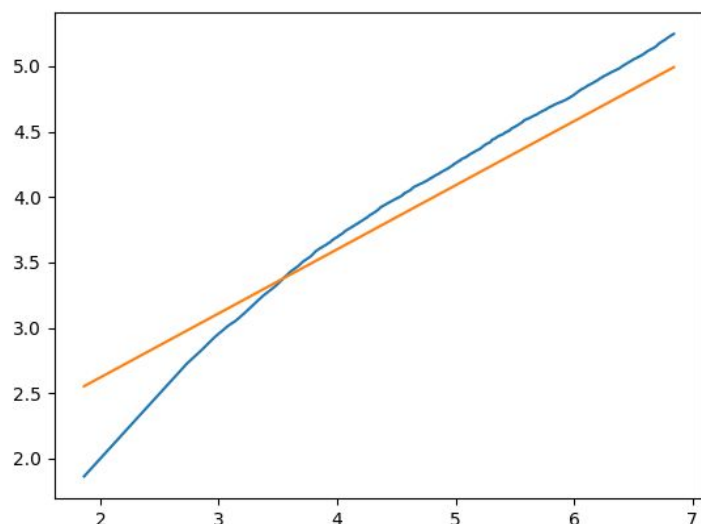
برای پیاده‌سازی این بخش به صورت ترکیبی از کتابخانه‌های `hazm` و `parsivar` استفاده کردیم بدین شکل که ابتدا کلمه را با `hazm` می‌دهم و سپس خروجی را به `parsivar` می‌دهم چون هر کدام از این کتابخانه‌ها یک سری از کلمات را نمی‌توانستند ریشه‌یابی کنند اما ترکیب آن‌ها تا حدی خوبی متناسب با کلمات و نیاز ما بود لیست کلماتی که با ریشه‌یابی به یکی از کلمات ذکر شده تبدیل شده‌اند را می‌توانید در فایل `stemming_words.json` فایل‌های پروژه ببینید

## قوانین Zipf و Heaps:

برای قانون Heaps می‌توان گفت حالت دوم بیشتر از اول در این قانون صادق است که دلیل آن هم اشکال متفاوتی نوشتاری توکن‌ها در دیکشنری در حالت اول هست اما در حالت دوم با توجه به نمودارهای زیر می‌توان گفت به میزان خوبی قانون Heaps برقرار است و از طرف دیگر هم می‌توان گفت بهترین برازش خطی در حالت دوم خط  $y=0.49x+1.60$  هست که تا حد زیادی منطبق بر قانون Heaps هست

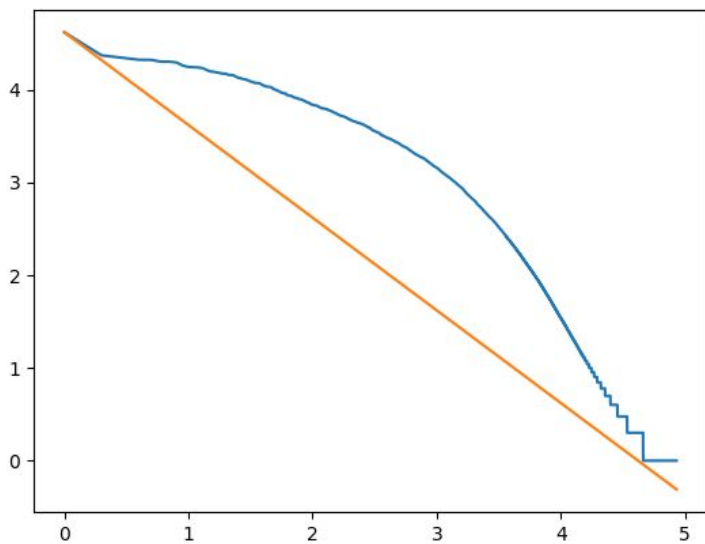


(حالت دوم)

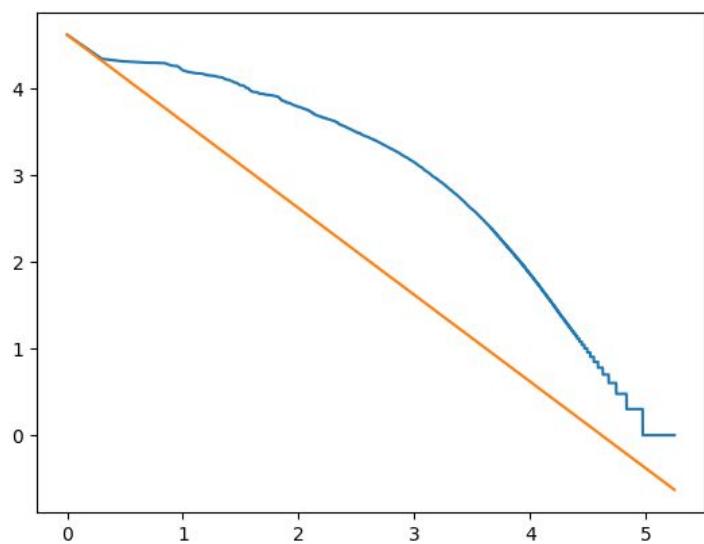


(حالت اول)

برای قانون Zipf می‌توان گفت که در هیچ کدام از یک حالت‌ها چندان برقرار نیست و فکر می‌کنم شاید به خاطر در نظر گرفتن مجموعه به نسبت بزرگی برای stopword باعث این اتفاق شده باشد چون حذف این کلمات باعث کاهش اندازه اولین کلمه یا همان  $K$  ثابت باشد و به همین دلیل در کل بالای نمودار تخمین Zipf باشیم



(حالت دوم)



(حالت اول)