

به نام خدا



دانشگاه تهران

دانشکده فنی

دانشکده مهندسی برق و کامپیوتر



درس پردازش زبان طبیعی

پاسخ تمرین ۲

نام و نام خانودگی: زهرا ریحانیان

شماره دانشجویی: ۸۱۰۱۰۱۱۷۷

اسفند ماه ۱۴۰۲

۳ پاسخ سوال اول
۳ پاسخ بخش اول
۴ term frequency – ساخت بردار جانیایی اول – پاسخ بخش دوم
۴ tf-idf – ساخت بردار جانیایی دوم – پاسخ بخش سوم
۵ ppmi – ساخت بردار جانیایی سوم – پاسخ بخش چهارم
۶ پاسخ بخش پنجم

پاسخ سوال اول

کد مربوط به این بخش در مسیر `codes/Q1.ipynb` موجود است.

پاسخ بخش اول

برای حل این سوال، ابتدا داده ها را از سایت Kaggle لود کردم و آن ها را در یک `data frame` ذخیره کردم. داده دارای دو کلاس ۰ و ۴ بود که آن را به ۰ و ۱ تغییر دادم. سپس با استفاده از متد `sample` از هر کلاس ۵۰۰۰ نمونه استخراج کرده و ذخیره نمودم.

مرحله بعد پیش پردازش های لازم را انجام دادم. ابتدا ستون `text` که همان توثیت ها بودند را به عنوان `X` و ستون `target` که کلاس هر توثیت را مشخص می کنند را به عنوان `y` ذخیره کردم. تابع `preprocess_twitter` کار پیش پردازش داده های `X` یا همان توثیت ها را انجام می دهد. هدف این است که داده ها را تمیز کنیم و تا جایی که می توان داده ها را غیر ضروری را حذف کنیم که از نظر زمان و حافظه بهینه عمل کنیم. در این تابع ابتدا تمام کلمات را حروف کوچک می کنیم چون زبان های برنامه نویسی به حروف حساس هستند و به طور مثال `The` و `the` را متفاوت در نظر می گیرند در حالی که می دانیم که یکی هستند.

در قدم بعدی در این تابع، کلمات توقف را حذف می کنیم. کلمات تکراری هستند که ارزش اضافی به بردار جانمایی متن نمی دهند و حذف آن ها باعث افزایش کارایی محاسبات و فضا می شود. با استفاده از کتابخانه `nlk` تمام کلمات توقف را بدست آورده و آن ها را حذف می کنیم.

بعد از این مراحل ایمیل ها، لینک ها و اعداد را حذف می کنیم چون اطلاعات زیادی راجع به مثبت یا منفی بودن یک توثیت نخواهند داد. در آخر هم علائم نگارشی را حذف می کنیم. علائم نگارشی از جمله نماد های غیر ضروری هستند اما در حذف آن ها مشکلاتی نیز وجود دارد مثل `U.S` که مخفف ایالات متحده است که تبدیل به `us` می شود. در اینجا با استفاده از `maketrans` یک جدول ترجمه ایجاد شد که به متد `str.translate` به عنوان ورودی داده شد و علائم نگارشی حذف گردید.

در متد دیگری کار `Stemming` روی داده انجام شد. `Stemming` یک تکنیک `normalization` متن است که داده های متن خام را به قالبی قابل خواندن برای کارهای پردازش زبان طبیعی تبدیل می کند. به طور خاص، این فرآیند کاهش شکل عطف یک کلمه به یک شکل به اصطلاح «`stem`» یا ریشه است که در زبان شناسی به عنوان «لم» نیز شناخته می شود. در انجام این کار، هدف `stemming` بهبود پردازش متن در سیستم های یادگیری ماشین و بازیابی اطلاعات است. `Stemming` باعث کاهش بعد بردار های جانمایی می شود و به فهم کلمات خارج از واژگان کمک می کند و بنابراین دقت مدل های آماری `NLP` را بهبود می بخشد.

در اینجا از `Porter stemmer` استفاده شد که یکی از محبوب ترین روش های ریشه یابی است که در سال ۱۹۸۰ ارائه شد. این روش بر این ایده استوار است که پسوندها در زبان انگلیسی از ترکیبی از پسوندهای کوچکتر و ساده تر ساخته شده اند. این `stemmer` به سرعت و سادگی معروف است. کاربردهای اصلی `Porter stemmer` شامل داده کاوی و بازیابی اطلاعات است.

در آخر این بخش بعد از پیش پردازش و ریشه یابی داده ها را همان طور که خواسته شد به دو قسمت test و train تقسیم شد.

پاسخ بخش دوم – ساخت بردار جانمایی اول – TERM FREQUENCY

در این بخش ۲ تابع اصلی داریم. تابع find_vocab واژگان را می یابد و یک توکن <UNK> هم اضافه می کنیم که توکن های خارج از واژگان را شامل می شود. تابع find_term_freq_matrix ماتریسی را که در صورت سوال توصیف شد، می یابد. ماتریسی که یک دیتافریم است. ستون های آن واژگان و سطر های آن توئیت ها هستند. هر سلول هم تعداد کلمه را نشان می دهد. بخشی از دیتافریم به صورت زیر است:

	dooooooooooooooooooooooooooooo	clubi	radiu	thk	scene	ole	wthe	rochest	academyquot	christin	...	carolina	impromptu	shatter	badminton	beachim	regal	obviou	smilequot	twitpic	indien
0		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	
7995		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
7996		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
7997		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
7998		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
7999		0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

8000 rows x 10388 columns

شکل ۱ بخشی از دیتافریم داده آموزش

هم برای داده آموزش و هم برای داده تست این دیتافریم محاسبه شد. برای داده تست هم به این صورت عمل کردم که ابتدا کلماتی از آن که در واژگانی که روی داده آموزش بدست آوردیم، وجود نداشت را حذف کردم و به جای آن ها توکن <UNK> قرار دادم و بعد دیتافریم term frequency را روی آن محاسبه کردم. در واقع واژگان آموزش را مرجع قرار دادم که ستون های داده آموزش و تست یکی باشند و برای کلاس بندی مشکل بوجود نیاید. با این کار مشکل کلمات خارج از واژگان را تا حدودی برطرف شد. از طرفی هم چون در ابتدا روی داده stemming انجام داده بودم، به حل این موضوع کمک می کرد.

پاسخ بخش سوم – ساخت بردار جانمایی دوم – TF-IDF

در این بخش با استفاده از فرمول هایی که در کلاس گفته شده بود ماتریس را تشکیل دادم. فرمول ها به شرح زیر بودند:

$$tf_{t,d} = \begin{cases} 1 + \log_{10} \text{count}(t,d) & \text{if } \text{count}(t,d) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$idf_t = \log_{10} \left(\frac{N}{df_t} \right)$$

ابتدا در تابع `find_word_count` برای هر یک از واژگان، تعداد توثیت هایی که شامل آن ها بودند را محاسبه کردم که محاسبه `idf` را سریع تر انجام دهم. در نهایت برای تک تک کلمات محاسبه `tf-idf` انجام شد و در یک دیتافریم که سطر آن توثیت ها و ستون های آن، واژگان را تشکیل می دادند، ذخیره شد. در زیر بخشی از ماتریس `tf-idf` مربوط به داده آموزش را مشاهده می کنید:

شکل ۲ بخشی از ماتریس `tf-idf` مربوط به داده آموزش

چون تعداد سطر ها و ستون ها زیاد است، نمایش تمام آن ممکن نبود. این ماتریس برای داده تست هم به صورت جداگانه محاسبه و ذخیره شد. در این بخش هم مانند بخش قبل ستون ماتریس داده های تست بر مبنای واژگان داده آموزش است به اضافه یک توکن `<UNK>` که کلماتی که در داده تست است ولی در داده آموزش نیست با این توکن جایگزین شدند.

پاسخ بخش چهارم - ساخت بردار جانمایی سوم - PPMI

در این بخش نیز با استفاده از فرمولی که در کلاس گفته شده بود، اقدام به محاسبه ماتریس خواسته شده کردم. فرمول آن به شرح زیر است:

$$PPMI(w, c) = \max(\log_2 \frac{P(w, c)}{P(w)P(c)}, 0)$$

این ماتریس هم همانند دو ماتریس بخش های قبل به این صورت است که سطر آن توثیت ها و ستون های آن، واژگان را تشکیل می دهند. این ماتریس برای هر دو داده آموزش و تست محاسبه شد و باز هم همانند دو بخش قبل ستون ماتریس داده های تست بر مبنای واژگان داده آموزش است به اضافه یک توکن `<UNK>` که کلماتی که در داده تست است ولی در داده آموزش نیست با این توکن جایگزین شدند.

بخشی از ماتریس `ppmi` داده آموزش را در این جا مشاهده می کنید:

