به نام خدا

تمرین سوم بازیابی اطلاعات

محمد ناصری

۸۱۰۳۰۰۴۸۶

دی ۱۴۰۰

بخش ۱ – Text Classification

هدف این تمرین، رده‌بندی احساسات Sentiment Classification است. برای این بخش، استفاده از کتابخانه‌های Scikit-learnو NLTK پیشنهاد میشود.

در ابتدا داده‌ها را با تکنیک‌هایی مانند حذف stop-word ها و stemming پیش پردازش میکنیم. مراحل پیش پردازش انجام شده در این قسمت به شرح زیر میباشد:

1. حذف تمامی کاراکترهای خاص
2. حذف تک کاراکترها
3. حذف تک کاراکترهای از آغاز سند
4. تبدیل چند فضای خالی متوالی به یک فضای خالی
5. تبدیل به حروف کوچک
6. Tokenization توسط NLTK و RegexTokenizer
7. در نهایت stemmize کردن و حذف stop-wordها

برای انجام مراحل فوق از دو کتابخانه NLTK و re استفاده میکنیم تا به کلمات پیش پردازش شده برسیم.

در قسمت بعدی با کمک کتابخانه Scikit کلمات اسناد را توسط CountVectorizer به ماتریسی با سطرهای اسناد و ستون‌های کلمات تبدیل میکنیم که هر درایه نشان دهنده تعداد مشاهده کلمه در سند است. برای انتخاب featureها در مرحله اول به صورت شهودی میتوان از یک سری از صفات مانند خوب، بد و ... استفاده کرد ولی برای بهتر شدن نتایج در این قسمت از کلماتی که بیشترین تکرار را در میان نظرات دارن استفاده میکنیم(max_features = 1500) و ۱۵۰۰ کلمه برتر را بعنوان ویژگی‌ها در نظر میگیریم.

ماشین‌ها برخلاف انسان‌ها نمی‌توانند متن خام را درک کنند. ماشین ها فقط می توانند اعداد را ببینند. به ویژه، تکنیک های آماری مانند یادگیری ماشینی فقط می توانند با اعداد سروکار داشته باشند. بنابراین، باید متن خود را به عدد تبدیل کنیم.

رویکردهای مختلفی برای تبدیل متن به شکل عددی مربوطه وجود دارد. مدل Bag of Words و مدل Word Embedding دو مورد از متداول ترین رویکردها هستند. در این تمرین از Bag of Words کلمات برای تبدیل متن خود به عدد استفاده می کنیم.

اسکریپت بالا از کلاس CountVetorizer از کتابخانه sklearn.feature_extraction.text استفاده می کند. برخی از پارامترهای مهم وجود دارد که باید به سازنده کلاس منتقل شود. اولین پارامتر پارامتر

max_features است که روی 1500 تنظیم شده است. این به این دلیل است که وقتی کلمات را با استفاده از رویکرد Bag of Words به اعداد تبدیل می کنید، تمام کلمات منحصر به فرد در تمام اسناد به ویژگی تبدیل می‌شوند. همه اسناد می توانند حاوی ده ها هزار کلمه منحصر به فرد باشند. اما کلماتی که فراوانی بسیار پایینی دارند به طور غیرعادی پارامتر مناسبی برای طبقه بندی اسناد نیستند. بنابراین ما پارامتر max_features را روی 1500 قرار می دهیم، به این معنی که می خواهیم از 1500 کلمه رایج به عنوان ویژگی برای آموزش طبقه بندی کننده خود استفاده کنیم.

پارامتر بعدی min_df است و روی 5 تنظیم شده است. این مربوط به حداقل تعداد اسنادی است که باید حاوی این ویژگی باشد. بنابراین ما فقط آن کلماتی را که در حداقل 5 سند وجود دارد، درج می کنیم. به طور مشابه، برای ویژگی max_df، مقدار 0.7 تنظیم شده است. که در آن کسری با درصدی مطابقت دارد. در اینجا 0.7 به این معنی است که ما باید فقط کلماتی را که حداکثر در 70٪ از کل اسناد وجود دارد، درج کنیم. کلماتی که تقریباً در هر سندی وجود دارند معمولاً برای طبقه بندی مناسب نیستند زیرا هیچ اطلاعات منحصر به فردی در مورد سند ارائه نمی دهند.

مانند هر مشکل یادگیری ماشینی تحت نظارت دیگری، ما باید داده های خود را به مجموعه های آموزشی و آزمایشی تقسیم کنیم. برای انجام این کار، از ابزار train_test_split از کتابخانه sklearn.model_selection استفاده می کنیم و مدل را به ۰.۸ داده آموزشی و ۰.۲ آزمایشی تقسیم میکنیم.

حال مدل بدست آمده را توسط الگوریتم‌های رده‌بندهای Logistic regression ، Naïve Bayes و SVM بر روی داده‌های آموزشی، train میکنیم و نتایج آنها را توسط معیارهای accuracy و precision و recall و f1-score ارزیابی میکنیم.

Multinomial Naïve Bayes

Category	Precision	Recall	F1-score	Support
0	0.84	0.84	0.84	**2496**
1	0.84	0.84	0.84	**2504**
Accuracy				**0.841**

Logestic-Regression

Category	Precision	Recall	F1-score	Support
0	0.86	0.85	0.86	**2496**
1	0.86	0.87	0.86	**2504**
Accuracy				**0.8594**

SVM

Category	Precision	Recall	F1-score	Support
0	0.86	0.85	0.86	**2496**
1	0.85	0.87	0.86	**2504**
Accuracy				**0.859**

در مرحله بعد به جای تعداد وقوع هر یک از کلمات انتخاب شده، ترکیبی از tfو idfرا به عنوان مقدار ویژگیها در نظر میگیریم.

رویکرد کیسه کلمات برای تبدیل متن به اعداد خوب عمل می کند. با این حال، یک ایراد دارد. به یک کلمه بر اساس وجود آن در یک سند خاص امتیاز می دهد. این واقعیت را در نظر نمی گیرد که ممکن است این کلمه در اسناد دیگر نیز فراوانی بالا داشته باشد. TFIDF این مشکل را با ضرب فرکانس عبارت یک کلمه در فرکانس معکوس سند حل می کند.

فرکانس به صورت زیر محاسبه می شود:

TF= (Number of Occurrences of a word)/(Total words in the document)

و فرکانس معکوس سند به صورت زیر محاسبه می شود:

IDF(word) = Log((Total number of documents)/(Number of documents containing the word))

مقدار TFIDF برای یک کلمه در یک سند خاص بیشتر است اگر دفعات وقوع آن کلمه در آن سند خاص بیشتر باشد اما در سایر اسناد کمتر باشد.

Multinomial Naïve Bayes

Category	Precision	Recall	F1-score	Support
0	0.85	0.84	0.84	**2496**
1	0.84	0.86	0.85	**2504**
Accuracy				**0.846**

Logestic-Regression

Category	Precision	Recall	F1-score	Support
0	0.88	0.85	0.87	**2496**
1	0.86	0.88	0.87	**2504**
Accuracy				**0.8594**

SVM

Category	Precision	Recall	F1-score	Support
0	0.87	0.86	0.86	**2496**
1	0.86	0.87	0.87	**2504**
Accuracy				**0.859**

مشاهده میشود که با استفاده از tf-idf نتایج ارزیابی مقداری بهتر میشوند که دلیل آن وزن‌دهی مناسب تر کلمات توسط این روش نسبت به روش tf میباشد.

برای بهتر شدن نتایج اقداماتی را میتوان در نظر گرفت که تعدادی از آنها در طول پیاده سازی مطرح شده اند. برای مثال حذف کلماتی که در نظرات زادی وجود دارند و در نظر گرفتن max_df = 0.7 و یا حذف کلماتی که در اسناد بسیار کمی دیده شده اند با min_df = 5. همچنین با افزایش تعداد ویژگیهای در نظر گرفته شده نتایج به دلیل در نظر گرفتن مجموعه‌ای بزرگتر از ویژگیها، مقداری دقیق‌تر میشوند برای مثال برای max_features=2000 داریم:

Multinomial Naïve Bayes

Category	Precision	Recall	F1-score	Support
0	0.85	0.84	0.85	**2496**
1	0.84	0.85	0.85	**2504**
Accuracy				**0.8482**

Logestic-Regression

Category	Precision	Recall	F1-score	Support
0	0.88	0.86	0.87	**2496**
1	0.87	0.89	0.88	**2504**
Accuracy				**0.8752**

SVM

Category	Precision	Recall	F1-score	Support
0	0.87	0.86	0.86	**2496**
1	0.86	0.87	0.87	**2504**
Accuracy				**0.8644**

در نهایت بهترین عملکرد در میان رده‌بندهای بررسی شده برای رده بند Logistic-Regression میباشد که از لحاظ تمامی معیارها نسبت به باقی رده‌بندها برتری دارد و نتایج با دقت بیشتری را تخمین میزند.

مکانیسم یادگیری بین این دو مدلNaïve Bayes و Logistic-Regression کمی متفاوت است، که در آن Naive Bayes یک مدل مولد و رگرسیون لجستیک یک مدل افتراقی است.

مدل مولد: Naive Bayes توزیع مشترک ویژگی X و هدف Y را مدل می‌کند و سپس احتمال posterior را که به صورت $P(y|x)$ داده می‌شود، پیش‌بینی می‌کند.

مدل افتراقی: رگرسیون لجستیک به طور مستقیم احتمال خلفی $P(y|x)$ را با یادگیری نگاشت ورودی به خروجی با به حداقل رساندن خطا مدل می‌کند.

بیز ساده تمام ویژگی ها را به صورت مشروط مستقل فرض می کند. بنابراین، اگر برخی از ویژگی‌ها در واقع به یکدیگر وابسته باشند (در مورد فضای ویژگی بزرگ)، پیش‌بینی ممکن است ضعیف باشد.

تقسیم‌های رگرسیون لجستیک دارای فضا به صورت خطی هستند و معمولاً حتی زمانی که برخی از متغیرها همبستگی دارند به خوبی کار می‌کنند.

بیز ساده: وقتی اندازه داده های آموزشی نسبت به تعداد ویژگی ها کوچک است، اطلاعات/داده های احتمالات قبلی به بهبود نتایج کمک می کند.

رگرسیون لجستیک: زمانی که اندازه داده‌های آموزشی نسبت به تعداد ویژگی‌ها کوچک است، از جمله منظم‌سازی مانند رگرسیون کمند و ریج می‌تواند به کاهش بیش‌برازش کمک کند و منجر به یک مدل تعمیم‌یافته‌تر شود.

از طرفی SVM سعی می‌کند بهترین حاشیه (فاصله بین خط و بردارهای پشتیبانی) را پیدا کند که کلاس‌ها را از هم جدا می‌کند و این خطر خطا روی داده‌ها را کاهش می‌دهد، در حالی که رگرسیون لجستیک اینطور نیست، در عوض می‌تواند مرزهای تصمیم متفاوتی با وزن‌های متفاوت داشته باشد. که نزدیک به نقطه بهینه هستند.

به طور کلی، معمولاً توصیه می‌شود ابتدا از رگرسیون لجستیک استفاده کنید تا ببینیم مدل چگونه کار می‌کند، اگر شکست خورد، می‌توانیم از SVM بدون kernel استفاده کنید (که به عنوان SVM با kernel خطی شناخته می‌شود). رگرسیون لجستیک و SVM با هسته خطی عملکرد مشابهی دارند اما بسته به ویژگی ها، ممکن است یکی از دیگری کارآمدتر باشد.

بخش ۲ – Topic Modeling

در این تمرین هدف پیادهسازی مدلسازی موضوعی PLSAبا بهکارگیری موضوع زمینه میباشد. مجموعه اسناد D={d1, d2, …., dN} به صورت ترکیبی از k موضوع θ1, θ2, … , θk و با استفاده از تابع احتمال لگاریتمی زیر مدل میشوند:

$$\log p(D \mid \Theta, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|d_i|} \log \left\{ \sum_{k=1}^{K} p(z_{i,j} = k \mid \pi_i) p(d_{i,j} = w \mid \theta_k) \right\}$$

با به کارگیری موضوع زمینه ثابت،تابع احتمال به صوریت زیر تغییر میکند که در آن، کلمه با احتمال λ از موضوع زمینه p(w|D) و با احتمال λ -1 از ترکیب K موضوع تولید میشود:

$$\log p(D \mid \Theta, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|d_i|} \log \left\{ \lambda p(d_{i,j} = w \mid D) + (1-\lambda) \sum_{k=1}^{K} p(z_{i,j} = k \mid \pi_i) p(d_{i,j} = w \mid \theta_k) \right\}$$

در پیادهسازی انجام شده از روش EM برای محاسبه استفاده شده است که در قسمت اول و بدون در نظر گرفتم زمینه از فرمولهای زیر استفاده میشود:

EStep

Probability that **w in doc d** is generated from **topic θ_j**

$$p(z_{d,w} = j) = \frac{\pi_{d,j}^{(n)} p^{(n)}(w \mid \theta_j)}{\sum_{j'=1}^{k} \pi_{d,j'}^{(n)} p^{(n)}(w \mid \theta_{j'})}$$

MStep

$$\pi_{d,j}^{(n+1)} = \frac{\sum_{w \epsilon V} c(w,d) * p(z_{d,w} = j)}{\sum_{w' \epsilon V} \sum_{w \epsilon V} c(w,d) * p(z_{d,w} = j)}$$

در مرحله اول برای تکمیل پیادهسازی، قسمت preProcessing را با کمک روشهای پیش پردازش قبلا ذکر شده (در سوال قبل) تکمیل میکنیم و از counterVectorizer برای تبدیل اسناد به ماترسی کلمه-سند و

همچنین mapهای کلمه به آیدی و آیدی به کلمه استفاده میکنیم. (برای سرعت بخشیدن به اجرا تنها ۱۵۰۰ کلمه برتر را در نظر گرفته شده است.

پس از اجرای برنامه خروجی مرحله اول برای تخمین کلمات topicها به صورت زیر است:

1. year percent oil immigration prices million new children gas like
2. soviet police union one died people saturday friday officials officers
3. percent central degrees high snow season much nation new expected
4. gorbachev new soviet rating percent california study national system states
5. bush campaign president dukakis fbi summit global hispanic people program
6. fire new dukakis scientists state two may three could also
7. percent rose black year new report businesses owned last prices
8. bank company new would barry million israel jewish official government
9. administration people roberts former year would years trade last one
10. state would people embassy saudi iraq two president waste government

برای محاسبه مدل زمینه لازم است تا در پیاده‌سازی تغییراتی انجام شود. هدف از استفاده از مدل پس‌زمینه θB این است که مدل‌های موضوعی متمایزتر شود. از آنجایی که θB احتمالات بالایی به کلمات غیر متمایز و غیر آموزنده می دهد، ما انتظار داریم که چنین کلماتی توسط θB محاسبه شوند و بنابراین مدل های موضوعی متمایزتر باشند. θB با استفاده از کل مجموعه C تخمین زده می شود:

$$p(w|\theta B) = \frac{\sum_{d \epsilon C} c(w,d)}{\sum_{w \epsilon V} \sum_{d \epsilon C} c(w,d)}$$

p(wjθB) در طول فرآیند تخمین بعدی تغییر نمی کند. پارامترهای دیگری که باید تخمین زده شوند عبارتند از:

$$\Lambda = \{\theta_j, \pi_{d,j} | d \epsilon C. 1 \leq j \leq k\}$$

و log-likelihood کالکشن C عبارت است از:

$$\log p(C|\Lambda) = \sum_{d \in C} \sum_{w \in V} [c(w,d) \times \log(\lambda_B p(w|\theta_B) + (1-\lambda_B) \sum_{j=1}^{\kappa} (\pi_{d,j} p(w|\theta_j)))]$$

با این پیشفرض تغییراتی که در کد لازم هست پیاده شوند به ضرح زیر میباشند:

- محاسبه فرکانس کلمات و در نظر گرفتن آن به عنوان مدل زبانی زمینه
- در نظر گرفتن lambda در قالب متغیر lam
- اضافه کردن محاسبه p(z = B) در Estep
- اضافه کردن محاسبات مدل زبانی زمینه در Mstep

محاسبات روش EM با احتساب مدل زمینه به صورت زیر تعریف میشوند:

Estep

Hidden Variable (=topic indicator): $z_{d,w} \in \{B, 1, 2, \ldots, k\}$

$$p(z_{d,w} = j) = \frac{\pi_{d,j}^{(n)} p^{(n)}(w \mid \theta_j)}{\sum_{j'=1}^{k} \pi_{d,j'}^{(n)} p^{(n)}(w \mid \theta_{j'})}$$

$$p(z_{d,w} = B) = \frac{\lambda_B p(w \mid \theta_B)}{\lambda_B p(w \mid \theta_B) + (1 - \lambda_B) \sum_{j=1}^{k} \pi_{d,j}^{(n)} p^{(n)}(w \mid \theta_j)}$$

MStep

Hidden Variable (=topic indicator): $z_{d,w} \in \{B, 1, 2, \ldots, k\}$

$$\pi_{d,j}^{(n+1)} = \frac{\sum_{w \in V} c(w,d)(1 - p(z_{d,w} = B)) p(z_{d,w} = j)}{\sum_{j'} \sum_{w \in V} c(w,d)(1 - p(z_{d,w} = B)) p(z_{d,w} = j')}$$

$$p^{(n+1)}(w \mid \theta_j) = \frac{\sum_{d \in C} c(w,d)(1 - p(z_{d,w} = B)) p(z_{d,w} = j)}{\sum_{w' \in V} \sum_{d \in C} c(w',d)(1 - p(z_{d,w'} = B)) p(z_{d,w'} = j)}$$

با اجرا کردن پیاده‌سازی جدید و در نظر گرفتن λ=0.9 نتایج زیر حاصل میشود:

1. films plant mrs cool ill plants area dog foster election

2. reforms reform study intelligence concluded occupied average fiscal survey illegal
3. grain experiments forecast wyoming premium primarily gallon highs southwest snow
4. waste guns films planning nbc directors faith immediate asset proposal
5. plains rockies wyoming snow emissions nbc mid brush ratings california
6. chemical houston remove nbc osha series nominee felt pennsylvania sometimes
7. awards october barry ticket production vocal history bombs operation immigration
8. gene scientists church tanks businesses article animal consumer kohlberg kravis
9. robb mexican strait experiments owners recruit syria guerrillas kentucky highs
10. bradley tanks farmers question gas quayle auto models behind trading

در مرحله بعد نتایج را برای$\lambda=0.3$ محاسبه میکنیم:

1. administration twice demand county move forces feel jackson another fair
2. july us dukakis management unless january capital tuesday governor presidential
3. inflation september education car boost california protection although atlantic modest
4. board children supported stock estimated henry comment five system program
5. fear left share sometimes seemed running line sale several expected
6. prison role bank john mexico sell best mass claimed americans
7. fear speech victims system see number children democratic others committee
8. campaign heart increases agreement prices local agreed bad county peace
9. fast negotiations noted senate bank overall person sometimes county heard
10. deputy seven george export mayor leadership shares support national white

با افزایش λ کلمات و موضوعات یافته شده بیشتر specific میشوند در حالیکه با λ کمتر موضوعات کلی تر و جامع تر میشوند. برای یافت موضوعات بهتر،مناسبت است تا ترکیبی بینابین از این ۲ داشته باشیم برای مثال λ در بازه 0.4 تا 0.6 باشد.