گزارش تمرین دوم

(ثمین مهدی زاده ۸۱۰۱۰۰۵۲۶)

در این تمرین به کمک Naïve Bayes بر روی دو دیتاست classification انجام شده است. دیتاست اول شامل SMS های ham و spam و دیتاست دوم برای تشخیص پیام های راست و دروغ مورد استفاده قرار میگیرد.
در این تمرین پس از دریافت دیتاست های موجود با استخراج ویژگی بر روی هر کدام یک طبقه بند آموزش داده می شود و سپس به کمک داده های تست این طبقه بند مورد ارزیابی قرار میگیرد.

۱)داده های HAM و SPAM

تحلیل تاثیر پیش پردازش ها بر روی طبقه بندی

پیش پردازش های صورت گرفته در این بخش شامل حذف punctuation،stemming، lemmatization و حذف stopword ها بود. در این بخش پیش پردازش های متفاوتی انجام شد که نتیجه آن به طور خلاصه در جدول زیر قابل مشاهده است.(نتایج کامل تر در نوت بوک موجود است)

نوع پیش پردازش	Accuracy	Macro-precision	Macro-recall	Macro-f1
-	98.30	98.39	94.23	96.18
Punc_removal	97.85	97.19	93.41	95.19
stemming	98.12	97.67	94.13	95.80
lemmatization	98.39	98.15	94.84	96.41
stopword	98.92	98.78	96.56	97.64
Punc_removal Lemmatization stopword	97.40	95.74	92.87	94.24
Lemmatization stopword	98.74	98.38	96.18	97.24

همان طور که مشاهده می شود حتی بدون انجام هیچ گونه پیش پردازشی مدل به خوبی عمل می کند و دقت خوبی دارد. اما اگر توجه شود حذف punctuation ها باعث شده دقت مدل کمی پایین بیاید به این معنی که علائم نگارشی می توانند در تشخیص نوع پیام موثر باشند. همچنین می توان دید حذف stopword در بهبود مدل موثر بوده است. lemmatization نیز از آنجا که از ریشه کلمات استفاده می کند generalization مدل را بالا برده و دقت خوبی به دست آمده اما stemming از آن جا که مانند lemmatization از دیکشنری استفاده نمی کند ممکن است گاهی اوقات ریشه کلمات را به درستی پیدا نکند و همان طور که جدول بالا نیز نشان می دهد استفاده از stemming حتی دقت مدل را به نسبت حالت بدون هیچ پیش پردازش کاهش داده

است. دو سطر آخر نیز بیان می کند اگر چند روش به تنهایی موثر واقع شوند لزوما ترکیب آن ها دقت بیشتری به ما نمی دهد.
از آن جا که از بین پیش پردازش های انجام شده حذف stopword بیشترین دقت را داشت از آن در ادامه استفاده شده است.

توضیح ویژگی های انتخاب شده برای هر طبقه بند و دلیل انتخاب آن

تمام پیش پردازش های بخش قبل به کمک روش bow انجام شده بود که نتایج بهترین آنها که مربوط به حذف stopword ها بود به صورت زیر است:

```
['stopword']
              precision    recall  f1-score   support

         ham     0.9897    0.9979    0.9938       965
        spam     0.9859    0.9333    0.9589       150

    accuracy                         0.9892      1115
   macro avg     0.9878    0.9656    0.9764      1115
weighted avg     0.9892    0.9892    0.9891      1115
```

می دانیم که روش bow بر اساس تعداد کلمات کار می کند و از آن جایی که برخی کلمات(مانند the)تعداد تکرار زیادی دارند این روش لزوما همیشه خوب جواب نمی دهد. به همین منظور روش tf-idf که فرکانس مستندات را نیز در نظر می گیرد امتحان شد. نتایج این روش به شرح زیر است:

```
              precision    recall  f1-score   support

         ham     0.9660    1.0000    0.9827       965
        spam     1.0000    0.7733    0.8722       150

    accuracy                         0.9695      1115
   macro avg     0.9830    0.8867    0.9274      1115
weighted avg     0.9705    0.9695    0.9678      1115
```

مشاهده می شود که در این جا برخلاف انتظار دقت کلی کمی پایین تر رفته اما مقدار precision برای spam و مقدار recall برای ham افزایش یافته است. این پایین رفتن دقت می تواند به این علت باشه که مدل های naive bays که در کد به کار رفته با count بهتر کار می کنند.

علاوه بر این از آن جا که دیدیم punctuation ها در طبقه بندی مفید هستند ویژگی هایی مانند طول جمله، تعداد علامت سوال و علامت تعجب نیز مورد بررسی قرار گرفتند. هنگامی که به تنهایی از این سه ویژگی استفاده شد نتایج زیر به دست آمد:

	precision	recall	f1-score	support
ham	0.8704	0.9948	0.9284	965
spam	0.5833	0.0467	0.0864	150
accuracy			0.8673	1115
macro avg	0.7268	0.5207	0.5074	1115
weighted avg	0.8317	0.8673	0.8152	1115

مشاهده می شود که حتی در این حالت و بدون استفاده از مدل bow می توانیم مدلی با دقت 86 درصد به دست آوریم. در صورتی که bow نیز به این سه ویژگی اضافه شود نتایج به صورت زیر به دست می آید:

	precision	recall	f1-score	support
ham	0.9777	1.0000	0.9887	965
spam	1.0000	0.8533	0.9209	150
accuracy			0.9803	1115
macro avg	0.9889	0.9267	0.9548	1115
weighted avg	0.9807	0.9803	0.9796	1115

می بینیم در حالتی که تنها از bow استفاده کرده بودیم دقت کمی بالاتر بود و ممکن است اضافه کردن ویژگی های جدید به دلیل بالابردن پیچیدگی دقت را کمی پایین آورده باشد اما رسیدن به دقت 86 در حالت بالا نشان می دهد که punctuation ها می توانند بسیار موثر باشند.

گزارش نتایج طبقه بندی و تحلیل آن

در بخش قبل مشاهده شد که بهترین مدل به دست آمده برای آموزش داده ها مدلی است که ویژگی های آن به کمک bow به دست آمده و برای پیش پردازش از حذف stopword ها استفاده می کند.

```
['stopword']
              precision    recall  f1-score   support

         ham     0.9897    0.9979    0.9938       965
        spam     0.9859    0.9333    0.9589       150

    accuracy                         0.9892      1115
   macro avg     0.9878    0.9656    0.9764      1115
weighted avg     0.9892    0.9892    0.9891      1115
```

تصویر بالا نشان می دهد که دقت کلی بر روی مدل حدود ۹۸ درصد است که دقت مناسبی است. همچنین می توان دید داده های موجود در کلاس ham کمی دقت بالاتری دارند و بهتر تشخیص داده شده اند. البته این اختلاف خیلی جزئی است که می تواند با مشاهده ی داده های بیشتر و یا قوی تر کردن مدل دقت این دسته را نیز بالاتر برد(و یا حتی دقت کلی)

۲) داده های sentimental LIAR

تحلیل تاثیر پیش پردازش ها بر روی طبقه بندی

پیش پردازش های انجام شده در این بخش نیز مانند بخش قبلی است. نتایج این پیش پردازش ها در جدول زیر قابل مشاهده است.

نوع پیش پردازش	Accuracy	Macro-precision	Macro-recall	Macro-f1
-	61.56	60.70	60.39	60.43
punc_removal	60.85	59.98	59.74	59.78
stemming	61.64	60.89	60.77	60.81
lemmatization	63.06	62.26	61.89	61.95
stopword	61.33	60.47	60.20	60.24
stemming stopword	60.77	59.94	59.75	59.79
stemming lemmatization	61.72	60.97	60.84	60.88

در این جا نیز مشاهده می شود حذف punctuation دقت را پایین آورده است. نکته ای که در اینجا قابل توجه است این است که lemmatization به خوبی توانسته دقت مدل را بالا ببرد. stemming نیز که در روش قبل دقت را به نسبت حالت بدون هیچ پیش پردازش پایین برده بود در این جا کمی آن را بهبود داده است. علاوه بر این حذف stopword که در قسمت پیشین به خوبی عمل میکرد در اینجا تاثیر چندانی نداشته و حتی کمی باعث کمبود دقت مدل شده است. بنابراین این می توان گفت بسته به متن پردازش شده ممکن است روش های مورد استفاده تاثیر متفاوتی بر روی دقت مدل بگذارند.
از آن جا که استفاده از lemmatization در بین حالت های امتحان شده بیشترین دقت را داشت در ادامه از آن استفاده می شود.

توضیح ویژگی های انتخاب شده برای هر طبقه بند و دلیل انتخاب آن

```
['lemmatization']
              precision    recall  f1-score   support

           0     0.5855    0.5262    0.5543       553
           1     0.6597    0.7115    0.6846       714

    accuracy                         0.6306      1267
   macro avg     0.6226    0.6189    0.6195      1267
weighted avg     0.6273    0.6306    0.6277      1267
```

تا این جا بهترین حالت به دست آمده به کمک ویژگی های استخراج شده از bow و با پیش پردازش lemmatization بود. لازم به ذکر است که در این جا عدد 0 به معنی false و عدد 1 به معنی برچسب true است.
مشابه قسمت قبل این بار نیز از روش tf-idf استفاده می شود و نتایج زیر به دست می آید:

```
              precision    recall  f1-score   support

           0     0.5924    0.3363    0.4291       553
           1     0.6149    0.8207    0.7031       714

    accuracy                         0.6093      1267
   macro avg     0.6036    0.5785    0.5661      1267
weighted avg     0.6051    0.6093    0.5835      1267
```

مجددا مشاهده می شود که این روش به نسبت bow که تنها با تعداد کار می کند برخلاف انتظار دقت پایین تری دارد که می تواند مربوط به توابع استفاده شده برای آموزش مدل باشد.

در ادامه سایر ستون های دیتاست بررسی شدند (علاوه بر ستون های اصلی طول پیام، تعداد علامت سوال و علامت تعجب نیز اضافه شدند) تا مشاهده شود کدام یک می تواند در تقسیم بندی مفید باشد. ماتریس همبستگی میان سایر ویژگی ها به شکل زیر است:

	label	sentiment	anger	fear	joy	disgust	sad	length	count?	count!
label	1.000000	0.043842	-0.060934	0.026140	0.016214	-0.050029	0.055575	0.041418	-0.006885	-0.023620
sentiment	0.043842	1.000000	-0.178887	-0.092071	0.349332	-0.152665	-0.200750	-0.056388	-0.029659	0.042919
anger	-0.060934	-0.178887	1.000000	0.077972	-0.398449	0.315868	-0.004062	0.028581	0.016014	0.017040
fear	0.026140	-0.092071	0.077972	1.000000	-0.243583	-0.116576	0.157458	-0.001854	0.006811	-0.011252
joy	0.016214	0.349332	-0.398449	-0.243583	1.000000	-0.392577	-0.328222	-0.014797	-0.024881	0.039898
disgust	-0.050029	-0.152665	0.315868	-0.116576	-0.392577	1.000000	-0.064410	0.060148	-0.011531	0.000827
sad	0.055575	-0.200750	-0.004062	0.157458	-0.328222	-0.064410	1.000000	0.069889	-0.008188	-0.039079
length	0.041418	-0.056388	0.028581	-0.001854	-0.014797	0.060148	0.069889	1.000000	0.025607	0.005626
count?	-0.006885	-0.029659	0.016014	0.006811	-0.024881	-0.011531	-0.008188	0.025607	1.000000	0.034767
count!	-0.023620	0.042919	0.017040	-0.011252	0.039898	0.000827	-0.039079	0.005626	0.034767	1.000000

جدول بالا نشان می دهد که اکثر این ویژگی ها همبستگی کمی با label که همان برچسب کلاس است دارند و بنابراین نمی توان انتظار داشت که تغییر زیادی در نتیجه بدهند. در صورتی که تمام ویژگی های ذکر شده در بالا را بدون استفاده از bow در مدل خود استفاده کنیم به دقتی حدود ۵۵ خواهیم رسید:

	precision	recall	f1-score	support
0	0.4907	0.4792	0.4849	553
1	0.6039	0.6148	0.6093	714
accuracy			0.5556	1267
macro avg	0.5473	0.5470	0.5471	1267
weighted avg	0.5545	0.5556	0.5550	1267

اگر ویژگی های bow را نیز به مدل اضافه کنیم دقت به دست آمده تقریبا برابر با ۶۰ درصد است که همچنان به نسبت حالتی که تنها از bow استفاده کنیم(حدود ۶۳ درصد) کمتر است. پیچیده شدن مدل می تواند یکی از عوامل پایین آمدن دقت باشد. برای مثال ماتریس همبستگی بالا نشان می دهد که میان ویژگی های اضافه شده anger بیشترین وابستگی را با برچسب داده ها دارد. اگر تنها این ویژگی را به مدل bow اضافه کنیم دقت بالاتری از حالت قبل به دست می آوریم. چرا که با افزودن ویژگی های زیاد ممکن است مدل به سمت حفظ داده های آموزش برود و generalization خود را از دست بدهد. نتایج زیر مربوط به مدلی است که تنها از anger در کنار bow استفاده می کند.(نتایج سایر ویژگی ها نیز در نوت بوک آورده شده که برای جلو گیری از تکرار بهترین آن در زیر آمده است)

anger

	precision	recall	f1-score	support
0	0.5648	0.5986	0.5812	553
1	0.6740	0.6429	0.6581	714
accuracy			0.6235	1267
macro avg	0.6194	0.6207	0.6196	1267
weighted avg	0.6264	0.6235	0.6245	1267

منطقی است که داشتن اطلاعات بیشتر در مورد احساسات در نتیجه گیری تاثیر گذار باشد و یا آن را بهبود ببخشد. برای مثال ممکن است انسان ها در هنگام ترس بیشتر دروغ بگویند و همین موضوع بتواند دسته بندی را قوی تر کند. اما شاید مدل ساده ای مانند naïve bayes نتواند به خوبی از این اطلاعات استفاده کند و به همین جهت است که اطلاعات اضافه شده خیلی تاثیری بر روی دقت نداشته اند. باید در نظر گرفت که استفاده از مدل های قوی تر به همراه این ویژگی ها احتمالا دقت بالاتری به دست می آورد.

گزارش نتایج طبقه بندی و تحلیل آن

با بررسی ویژگی های مختلف و روش های پیش پردازش بهترین مدل به دست آمده برای این قسمت مدل با پیش پردازش lemmatization و استفاده از روش bow برای ویژگی ها بود. نتایج این دسته بندی به شرح زیر است:

['lemmatization']

	precision	recall	f1-score	support
0	0.5855	0.5262	0.5543	553
1	0.6597	0.7115	0.6846	714
accuracy			0.6306	1267
macro avg	0.6226	0.6189	0.6195	1267
weighted avg	0.6273	0.6306	0.6277	1267

در مقاله[1] مرجعی که از دیتاست مشابه استفاده شده بود دقت مدل ۷۰ درصد بود. لازم به ذکر است که در این مقاله از مدل های پیچیده تری مانند bert استفاده شده بود اما در این جا تنها به کمک naïve bayes که ساده سازی های فراوانی را در نظر می گیرد دقت بالا به دست آمد. در این جا نیز می توان گفت که مدل به دست آمده در تشخیص پیام های واقعی(true) بهتر عمل کرده و شاخص های آن دارای اعداد بالاتری هستند.

[1] https://arxiv.org/abs/2009.01047

نحوه اجرا:
برای اجرای کد های این تمرین کافی است فایل های مربوط به دیتاست در کنار فایل نوت بوک قرار گیرد.