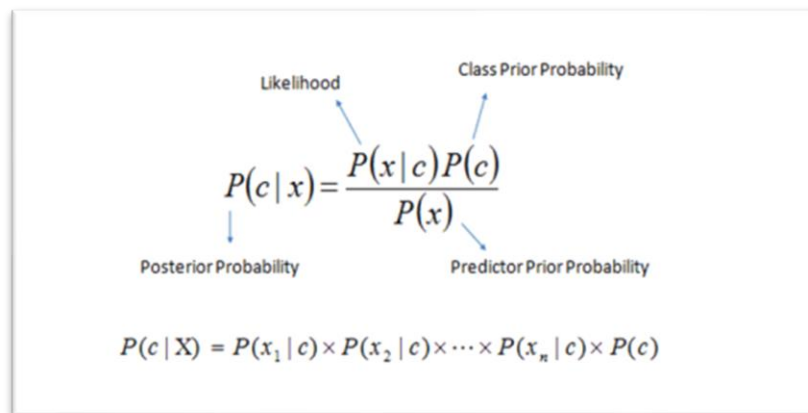


به نام خدا

پروژه سوم هوش مصنوعی

نرگس غلامی ۸۱۰۱۹۸۴۴۷



The diagram shows the Bayes' Theorem formula with labels for its components:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Labels and arrows:

- Likelihood** points to $P(x|c)$
- Class Prior Probability** points to $P(c)$
- Posterior Probability** points to $P(c|x)$
- Predictor Prior Probability** points to $P(x)$

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

هدف پروژه: هدف این پروژه تجزیه و تحلیل آگهی سایت دیوار و دسته بندی مناسب آن ها می باشد. از طرف دیگر تصمیم داریم با استفاده از داده های عنوان و توضیحات هر آگهی، دسته بندی آن را پیش بینی نماییم.

توضیح کلی پروژه: در ابتدا به نرمال کردن داده ها می پردازیم و سپس در مرحله ی بعد با استفاده از مفهوم bags of word تابع بررسی احتمال یک دسته خاص بودن به شرط بودن یک کلمه در آگهی را می نویسیم و سعی می کنیم این مدل را بهبود دهیم. صحت این تابع را بررسی می نماییم و در آخر از معیارهای مختلف برای ارزیابی توابع خود استفاده می کنیم.

در مرحله ی اول داده های دو فایل divartrain و divartest را برای نرمال کردن می خوانیم. داده ی divartrain در جهت محاسبه احتمال است که روی داده های فایل divartrain تست می شوند.

برای نرمال کردن داده ها از حذف ایست واژه ها کمک گرفته می شود. ایست واژه های معروف در یک فایل تحت عنوان stopwords2 ریخته می شود و سپس اگر توضیحات یا عنوان شامل آن ایست واژه ها باشند حذف می شود. یک عملیات دیگر که برای نرمال کردن داده ها استفاده شد عملیات جایگذاری برخی علائم بود که به صورت پایین نوشته شده است.

```
def placement(words):
    for i in range(len(words)):
        words[i] = words[i].replace( "\n","")
        words[i] = words[i].replace( "\t","")
        words[i] = words[i].replace( "...","")
    return words
```

سوال یک: یکی دیگر از روش های معمول normalization ، stemming یا lemmatization می باشد. توابع این دو کتابخانه هضم موجود می باشد و پیاده سازی شده است.

```
def stem(words):
    stemmer = Stemmer()
    words = [stemmer.stem(word) for word in words]
    return words

def lemmatize(words):
    lemmatizer = Lemmatizer()
    words = [lemmatizer.lemmatize(word) for word in words]
```

در کل این دو عمل به عنوان ریشه یابی شناخته می شوند.

عمل ریشه یابی این امکان را می دهند که فرم های مختلف یک کلمه را به یک فرم واحد تبدیل کنیم ، با این کار تعداد ویژگی ها کمتر می شود و همچنین شکل های مختلف یک کلمه حذف شده و کامپیوتر می تواند شکل های مختلف یک کلمه را یکی در نظر بگیرد.

رای ریشه یابی کلمات معمولاً از دو روش Stemming و Lemmatization استفاده می شود که هر دو روش قادر هستند ریشه ی یک کلمه را به دست بیاورند. برای مثال کلمه ای مانند "آمدن" را تصور کنید که ممکن است در جمله های مختلف به شکل های گوناگون ظاهر شود. برای مثال:

- دوستم به خیابان اصلی آمد

- آن‌ها از گردش آمدند.
- کاش دوستام هم آمده بودند.

تمامی فعل‌های "رفتن" در جملات بالا را می‌توان به کلمه‌ی آمد نگاشت کرد .

الگوریتم‌های مختلفی جهت انجام عمل Stemming (که یکی از روش‌های به دست آوردن ریشه‌ی کلمات است) وجود دارد. در زبان انگلیسی الگوریتم Porter بسیار معروف است. این الگوریتم طبق یک سری قاعده‌ی منظم (مثلاً حذف حرف S در آخر کلمات جمع) می‌تواند ریشه‌ی کلمات را با دقت خوبی به دست آورد. عمل Lemmatization نیز می‌تواند توسط روش‌هایی انجام شود. در این عمل نیاز است که از یک فرهنگ لغت یا چیزی شبیه به آن برای به دست آوردن ریشه‌ی لغات استفاده شود، چون عموماً روش‌های Lemmatization به صورت با قاعده نیستند.

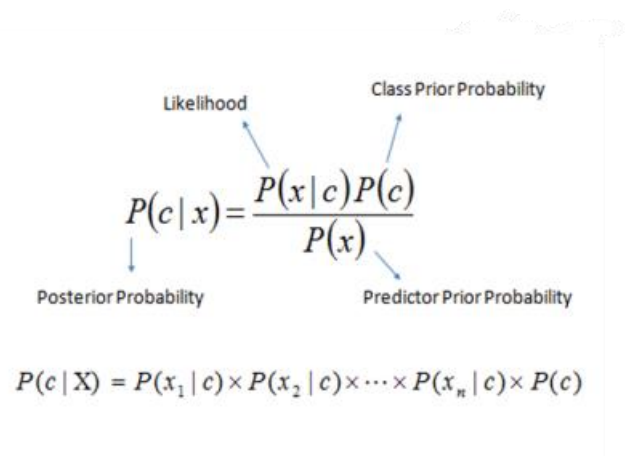
سوال دو:

Posterior probability در حقیقت برابر است با این که اگر بدانیم یک کلمه X را داریم چقدر احتمال دارد که در دسته‌بندی C باشیم.

Likelihood در این مسئله به ما می‌گوید اگر دانیم در دسته‌بندی C هستیم چقدر احتمال دارد که کلمه X وجود داشته باشد (یا انتخاب شود)

Prior probability یعنی احتمال این که در این دسته بندی باشیم نسبت به این که در دسته‌بندی‌های دیگر باشیم چقدر است.

Evidence یعنی احتمال انتخاب شدن یک کلمه بین همه‌ی کلمه‌ها



مفاهیم هر کدام از موارد بالا :

در کل قضیه بیز، میزان باور ما نسبت به یک پدیده را قبل و بعد از مشاهده شواهدی در تایید یا انکار آن پدیده به هم پیوند می‌زند.

Posterior probability میزان باور ما نسبت به C پس از مشاهده X

Likelihood/ Evidence میزان حمایت X از C می‌باشد.

Prior probability میزان باور ابتدایی ما به C می‌باشد.

من برای محاسبه این احتمال از هر دو ستون عنوان و توضیحات استفاده کردم.

سوال سوم:

کلمه‌ای مانند شی را در نظر بگیرید که سه معنی مختلف دارد. شیر جنگل، شیر خوردنی و شیر آب.

حال مثال‌های زیر را نگاه کنید:

نرگس شیر آب را بست.

حامد از شیرهای باغ وحش می‌ترسد.

فاطمه شیر و کیک را به عنوان صبحانه دوست دارد.

در همه‌ی مثال‌های بالا کلمه‌ی شیر استفاده شده است که اگر به معنی آن‌ها دقت نشود همه در یک گروه می‌افتند در حالی که معای متفاوتی دارند. در مثال‌های بالا استفاده از bigram کار را درست‌تر پیش می‌برد.

ولی فرضا در مثال‌های زیر حتی ممکن است به 3gram هم نیاز پیدا بکنیم.

فاطمه مقید به نوشیدن روزانه‌ی شیر ، آب و آبمیوه است.

حامد همیشه حواسش است که شیر آب آشپزخانه را ببندد.

در هر دو مثال بالا شیر و آب وجود دارد ولی این که معنی این مربوط به کدام کلمه است نیاز به یک کلمه‌ی سوم دارد.

سوال چهارم:

در مدل bags of words ما کلمات را مستقل از یکدیگر در نظر می‌گیریم و این احتمالات را در یکدیگر ضرب می‌کنیم. در صورتی که احیاناً یکی از این عبارات صفر باشد کل احتمال پسین را صفر می‌کند که این اتفاق، اتفاق خوبی نیست زیرا ممکن است کلمات دیگر جمله احتمال بالایی داشته باشند ولی در صورتی که تنها یک کلمه در آن دسته خاص وجود نداشته باشد کل احتمال صفر می‌شود. حتی ممکن است این کلمه در هیچ کدام از دسته‌ها موجود نباشد که در این صورت هیچ ایده‌ای نیز نداریم این جمله ممکن است متعلق به کدام دسته باشد. باز هم یک مشکل دیگر که وجود دارد این است که یک کلمه فقط در یک دسته‌بندی وجود داشته باشد که شاید هم لزوماً ربطی به داده‌ی اصلی ما نداشته باشد ولی احتمال ما به صورت قطعی نظر می‌دهد که این داده متعلق به آن دسته می‌باشد.

سوال پنجم:

$$P(w|c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V| + 1}$$

فرمول Additive Smoothing را مشاهده می‌نمایید. فرق این فرمول با فرمول قبلی این است که صورت و مخرج هر دو با یک جمع شده‌اند و همچنین مخرج با V جمع شده که V تعداد کلمات یونیک هر دسته‌بندی می‌باشند. همچنین $\text{count}(w, c)$ نیز به معنی تعداد کلمات w در دسته‌بندی c می‌باشد و $\text{count}(c)$ نیز تعداد کل کلمات دسته‌بندی می‌باشد. این فرمول به ما امکان تخصیص احتمالات غیر صفر را به کلماتی که در نمونه وجود ندارند، می‌دهد. انگار که این کلمه یک کلمه ناشناخته محسوب می‌شود و به کلمات یونیک ما اضافه می‌شود. این کار مشکلاتی که در جواب سوال پیش ذکر شد را برطرف می‌کند.

در این قسمت از تکه کد زیر برای کشیدن پلات‌ها استفاده می‌شود:

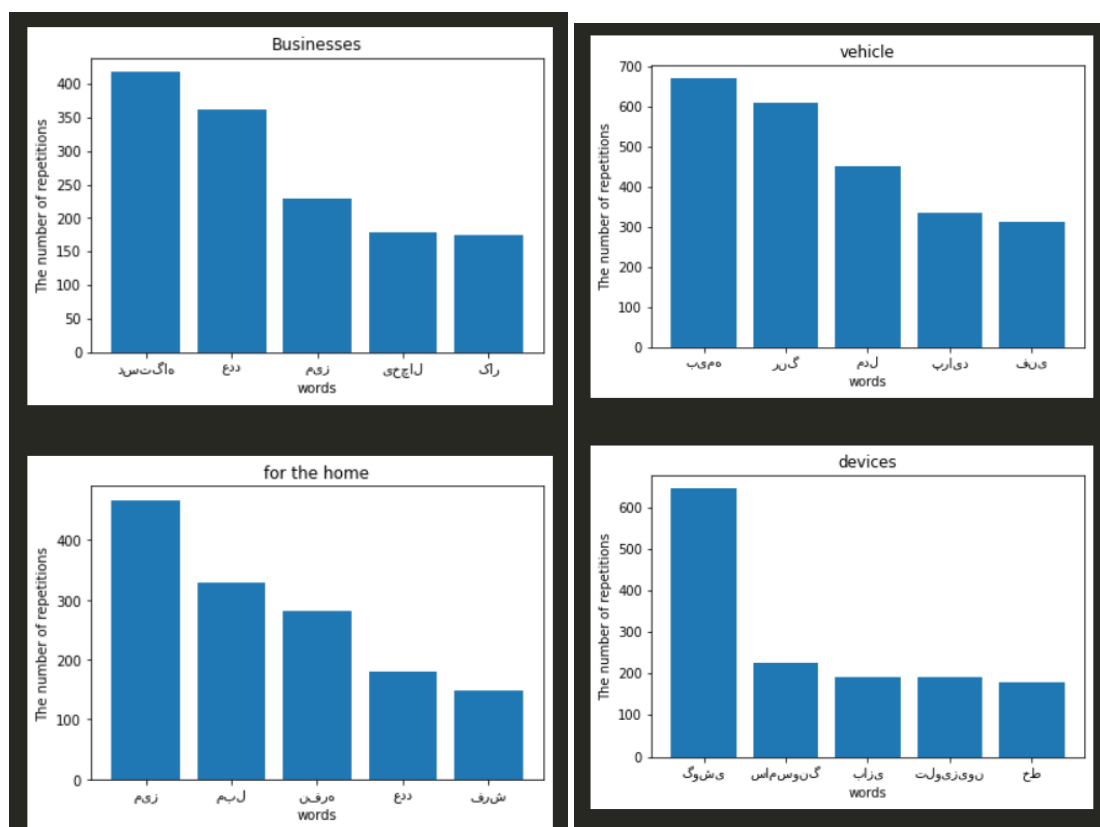
```
1 def plotCategory(category):
2     cat = deepcopy(category)
3     categorySort = sorted(cat.items(), key=lambda x: x[1], reverse=True)
4     c1 = [i[0] for i in categorySort]
5     c2 = [i[1] for i in categorySort]
6
7     DuplicateWords = set(hazm.stopwords_list("Duplicate_words.txt"))
8     count = 0
9     while(count < 5):
10         if c1[count] in DuplicateWords:
11             c1.remove(c1[count])
12             c2.remove(c2[count])
13         else:
14             count += 1
15
16     plt.xlabel("words")
17     plt.ylabel("The number of repetitions")
18     plt.plot(c1[:5], c2[:5])
19     plt.show()
```

✓ 0.2s

ابتدا دسته‌بندی مورد نظر برحسب تعداد تکرارشان سورت می‌شوند. سپس کلمات پرتکرار از صدر لیست‌های داده شده توسط تابع سورت حذف می‌شوند. این کلمات تکراری را خودم با بررسی اینکه یک کلمه ممکن است چقدر به یک دسته‌بندی مربوط باشد پیدا کردم و در یک فایل قرار دادم که شامل کلمات زیر می‌باشند:

تخفیف، فروش، کاملاً، نو، حد، تماس، سالم، تمیز، مناسب، همراه، قیمت، فوری، لطفاً، سلام

سپس ۵ کلمه با تعداد بالاتر انتخاب می‌شوند و پلات آن‌ها رسم می‌شود.



در هر پلات ۵ کلمه‌ی پرتکرار آن دسته آمده است

Vehicle: بیمه، رنگ، مدل، پراید، فنر

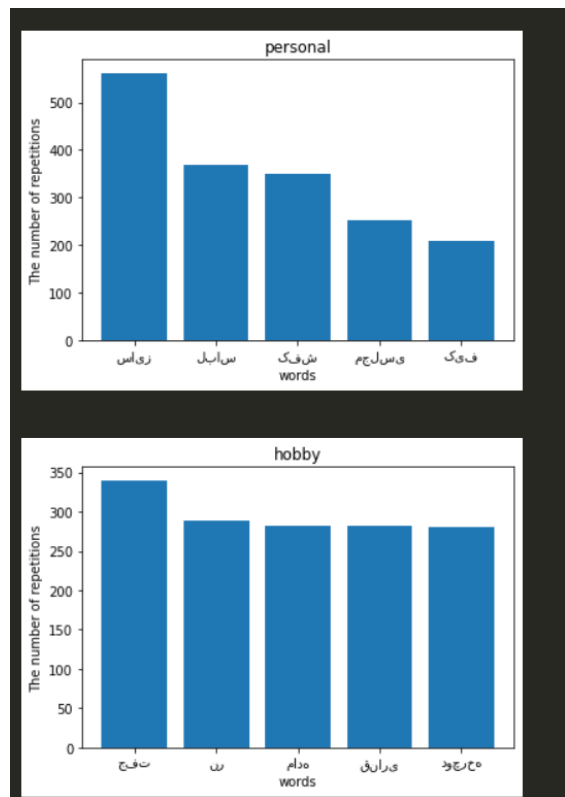
Devices: گوشی، سامسونگ، بازی، تلویزیون، خط

Business: دستگاه، عدد میز، یخچال، کار

For the Home: میز، مبل، نفره، عدد، فرش

Personal: سایز، لباس، کفش، مجلسی، کیف

Hobby: جفت، نر، ماده، قناری، دوچرخه



سوال هفت:

اول باید گفت (Accuracy) نقطه‌ی ضعف دارد، و آن این است که این معیار، نمی‌تواند تفاوتی بین خطای False Negative و خطای False Positive داشته باشد.

برای غلبه بر این مشکل دو معیار دیگر معرفی شده می‌کنیم که یکی از آن‌ها معیار صحت (Precision) است که فرمول آن را در صورت پروژه می‌بینیم. اگر با دقت به فرمول نگاه بکنیم، متوجه می‌شویم که تمرکز اصلی این معیار، بر روی درستی تشخیص‌های «بلی» توسط الگوریتم است. در واقع معیار صحت (Precision) معیاری است که به ما می‌گوید الگوریتم چند درصد «بلی»‌هایش درست بوده است. زمانی که ارزش false positive بالا باشد این معیار مناسب است ولی در اینجا لزوماً به کار می‌نمی‌آید.

یکی از مواردی که مدل با این معیار خوب کار نمی‌کند فرضاً تشخیص بیماران مبتلا به کوید ۱۹ می‌باشد. فرض کنید مقدار precision پایین باشد و الگوریتم تشخیص بیماری را بدهد. در این صورت از طرفی به بیمار استرس وارد می‌شود و از طرف دیگر مجبور است هزینه‌های گزاف بپردازد.

اما معیار کاربردی دیگری به نام پوشش (Recall) نیز وجود دارد و همان‌طور که از نامش پیداست، به دنبال محاسبه‌ی پوشش بر روی کل داده‌هاست. همان‌طور که مشاهده می‌کنید، تمرکز اصلی معیار پوشش (Recall) بر خلاف معیار صحت (Precision) بر روی داده‌هایی است که واقعاً «بلی» بوده‌اند. این مورد وقتی به کار می‌آید که True negative های بیشتری داشته باشیم که باز هم لزوماً در این مسئله به کار نمی‌آید. فرض کنید مدلی برای تشخیص بیماری کشنده وبا داشته باشیم. اگر این مدل recall پایینی داشته باشد اتفاقی که می‌افتد این است که تعداد افرادی که سالم در نظر می‌گیریم زیاد خواهد شد و فاجعه به بار خواهد آمد زیرا این بیماری ویروسی است.

سوال هشت:

معیار سومی به اسم معیار F1 هم وجود دارد که در واقع ترکیبی از معیارهای دقت و صحت است و می تواند در مواردی که هزینه False Positive و False Negative متفاوت است به کار رود.

برای محاسبه ی آن از میانگین گیری همساز یا Harmonic mean استفاده شده است. در مواردی که شامل نرخ ها و نسبت ها است، میانگین هارمونیک صحیح ترین مقدار میانگین را به ما می دهد مفهوم میانگین با مفهوم تصاعد بستگی نزدیک دارد. در تصاعد حسابی، هر جمله (به جز جمله ی اول و جمله ی آخر) میانگین حسابی دو جمله مجاور خود است. همچنین در تصاعد هندسی هر جمله (به جز جمله های اول و آخر) میانگین هندسی دو جمله مجاور خود است. به همین ترتیب می توان تصاعد همساز (یا تصاعد توافقی) را تعریف کرد؛ هر جمله تصاعد همساز (به جز دو جمله اول و آخر)، میانگین همساز دو جمله مجاور خود است. این مفهوم با فرمول ما که میانگینی از هر دوی precision و recall است تطابق دارد.

این کار به این علت برای ما اهمیت دارد که ما می خواهیم معیاری بینابین recall و precision داشته باشیم پس از این مدل میانگین گیری استفاده می نماییم.

سوال نه:

هنگامی که ما در مسئله ی خود دو متغیر داشته باشیم کار ما راحت است ولی هنگامی که تعداد این کلاس ها بیشتر می شود نیاز به یک معیار جدید داریم. در این سوال سه مورد را توضیح می دهیم.

Macro: این نوع از میانگین گیری همان میانگین گیری معمولی می باشد. یعنی برای تک تک متغیرها فرضا مقدار F1 حساب می شود و سپس جمع می شوند و تقسیم بر تعداد کل می شوند.

Weighted: در این میانگین گیری از مفهوم میانگین گیری وزن دار کمک می گیریم. یعنی جمع هر دسته ضربدر سهمش را حساب می کنیم و بعد تقسیم بر تعداد سهم ها می کنیم.

Micro: در این قسمت recall و precision کلاس ها را محاسبه می نماییم و بعد اگر این دو با هم برابر بودند مقدار Micro نیز برابر همین دو می باشد.

سوال ده:

الف) نتایج با استفاده از Additive Smoothing

	Businesses	Electronic Devices	For the home	Leisure Hobbies	Personal	Vehicles	All Classes
Precision	۰.۷۷	۰.۹۲	۰.۷۳	۰.۹۲	۰.۸۷	۰.۹۳	-
Recall	۰.۷۳	۰.۸۹	۰.۸۸	۰.۸۲	۰.۸۸	۰.۹۰	-
F1-score	۰.۷۵	۰.۹	۰.۸۰	۰.۸۷	۰.۸۷	۰.۹۳	-
Accuracy	-	-	-	-	-	-	۰.۸۵۴
Macro Avg	-	-	-	-	-	-	۰.۸۵۱
Micro Avg	-	-	-	-	-	-	۰.۸۵۴
Weighted Avg	-	-	-	-	-	-	۰.۸۵

```

47
48 print("F1: ", F1)
[92] ✓ 0.1s Python
... Accuracy is 0.8544444444444445
precision: {'vehicles': 0.9347079037800687, 'electronic-devices': 0.9238754325259516, 'businesses': 0.775438596491228, 'for-the-home': 0.7307692307692307, 'personal': 0.8721311475409836, 'leisure-hobbies': 0.924812030075188}
recall: {'vehicles': 0.9066666666666666, 'electronic-devices': 0.89, 'businesses': 0.7366666666666667, 'for-the-home': 0.8866666666666667, 'personal': 0.8866666666666667, 'leisure-hobbies': 0.82}
F1: {'vehicles': 0.9204737732656515, 'electronic-devices': 0.9066213921901528, 'businesses': 0.7555555555555555, 'for-the-home': 0.8012048192771084, 'personal': 0.8793388429752066, 'leisure-hobbies': 0.8692579505300353}

```

ب) بدون آن:

	Businesses	Electronic Devices	For the home	Leisure Hobbies	Personal	Vehicles	All Classes
Precision	۰٫۷	۰٫۵۵	۰٫۸	۰٫۲۱	۰٫۷۴	۰٫۸۱	-
Recall	۰٫۲۳	۰٫۱۸	۰٫۲۵	۰٫۹۵	۰٫۲۴	۰٫۲۵	-
F1-score	۰٫۲۵	۰٫۲۷	۰٫۳۸	۰٫۵۵	۰٫۲۷	۰٫۳۹	-
Accuracy	-	-	-	-	-	-	۰٫۳۵
Macro Avg	-	-	-	-	-	-	۰٫۳۶
Micro Avg	-	-	-	-	-	-	۰٫۳۶۱
Weighted Avg	-	-	-	-	-	-	۰٫۳۵۸

```

Accuracy is 0.35833333333333334
{'vehicles': 0.7835051546391752, 'electronic-devices': 0.6746987951807228, 'businesses': 0.7291666666666666, 'for-the-home': 0.7522935779816514, 'personal': 0.76, 'leisure-hobbies': 0.21673003802281368}
{'vehicles': 0.25333333333333335, 'electronic-devices': 0.18666666666666668, 'businesses': 0.23333333333333334, 'for-the-home': 0.2733333333333333, 'personal': 0.25333333333333335, 'leisure-hobbies': 0.95}
{'vehicles': 0.3828715365239295, 'electronic-devices': 0.2924281984334204, 'businesses': 0.3535353535353536, 'for-the-home': 0.4009779951100244, 'personal': 0.38, 'leisure-hobbies': 0.35294117647058826}
Macro F1: 0.36045904334555273
Weighted F1: 0.36166057349003783
Micro F1 0.35833333333333334

```

سوال یازده:

هنگامی که از Additive Smoothing برای محاسبه احتمال استفاده کردیم، مقدار تمام معیارها از جمله accuracy و F1 به طور قابل توجهی افزایش داشت که این نشان از این دارد که مدل Additive Smoothing مدل محاسبه‌ی احتمال مناسب‌تری می‌باشد.


```

personal
2 ['کشوی', 'مخفی', 'شیک'] for-the-home
electronic-devices
6 ['بردش', 'یه', 'تغییر', 'یافته', 'در'] leisure-hobbies
businesses
14 ['$NUM' اطلاعات', 'بیشتر در تلگرام'] leisure-hobbies
for-the-home
16 ['سیپوراکس', 'میکرو', 'مک', 'جی', 'ال', 'سرامیک', 'سرا', 'کاهنده', 'نیترات', 'نیتريت', 'اکواریوم', 'درجه'] leisure-hobbies
for-the-home
21 ['سایز ۴۲-۴۴', 'درختو', 'رنگ', 'بادمجونی', 'تاپ', 'زدگی', 'وبارگی'] personal

```

این چند مورد اشتباه تشخیص داده شدند. یکی از علل این بود که کلمات کلی حذف نشده‌اند. به طور مثال کلمه شیک بیشتر در دسته personal تکرار شده ولی این آگهی مربوط به for the home بوده است. در حالی که کلمه‌ی شیک خود بایاس خاصی نسبت به دسته‌ی خاصی را اعلام نمی‌کند. بعضی کلمات انگلیسی مناسب‌تر بود که به فارسی تبدیل شوند. یا اعداد از حروف جدا شوند. این‌ها در حقیقت دسته بندی را مناسب‌تر خواهد کرد در حقیقت ما در این مدل خود کلمات بدون بایاس را حذف نکرده‌ایم. این یکی از مشکلات عمده می‌باشد.

پیشنهادهایی برای بهبود پروژه: پروژه مناسب بود ممنونم.

منابع:

<https://dataio.ir/%D9%BE%DB%8C%D8%B4-%D9%BE%D8%B1%D8%AF%D8%A7%D8%B2%D8%B4-%D9%85%D8%AA%D9%88%D9%86-%D9%81%D8%A7%D8%B1%D8%B3%DB%8C-%D8%A8%D8%A7-%D8%A7%D8%B3%D8%AA%D9%81%D8%A7%D8%AF%D9%87-parsivar-cmug6xkqbl6d>

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1>

<https://chistio.ir/%DB%8C%D8%A7%D9%81%D8%AA%D9%86-%D8%B1%DB%8C%D8%B4%D9%87-%DA%A9%D9%84%D9%85%D8%A7%D8%AA-stemming-lemmatization>

لکچر آمار و احتمال استاد بهرک

https://en.wikipedia.org/wiki/Harmonic_mean

<https://chistio.ir/precision-recall-f>

<https://bigdata-ir.com/%D8%B1%D9%88%D8%B4-%D9%87%D8%A7-%D9%88-%D9%85%D8%B9%DB%8C%D8%A7%D8%B1%E2%80%8C%D9%87%D8%A7%DB%8C-%D8%A7%D8%B1%D8%B2%DB%8C%D8%A7%D8%A8%DB%8C-%D8%A7%D9%84%DA%AF%D9%88%D8%B1%DB%8C%D8%AA%D9%85-%D9%87>

https://fa.wikipedia.org/wiki/%D9%85%DB%8C%D8%A7%D9%86%DA%AF%DB%8C%D9%86_%D9%87%D9%85%D8%B3%D8%A7%D8%B2