حسین لشگری – 9731055 گزارش پروژه چهارم

نحوه مدل سازی :

طبق مراحل گفته شده، توابعی تعریف شدند که به ترتیب در ادامه شرح داده می شوند.

توضیح توابع :

- **preprocess(datalist)** : این تابع یک لیست شامل کلمات را می گیرد و کلماتی که دارای کاراکتر های بی تاثیر هستند را حذف می کند.

- **make_dict(dataset, mode)** : این تابع یک متن را می گیرد و از آن دیکشنری شامل تک کلمات و تعدادشان و همینطور دیکشنری شامل جفت کلمات و تعدادشان را می سازد و بر می گرداند. به این صورت که ابتدا با توجه **mode** مشخص می شود که کلمات پرتکرار از دیکشنری حذف شوند یا خیر. سپس کلمات در متن خوانده می شوند و به دیکشنری اضافه می شوند.

- **bigram(uni_dict, bi_dict, word1, word2)** : این تابع احتمال اینکه کلمه **word2** بعد از **word1** بیاید را حساب می کند.
طبق روش مشخص شده و تعیین لامبدا 1 تا 3 و شمردن تعداد از دیکشنری احتمال مشخص می شود. لامبدا ها تاثیر به سزایی در **precision** و **recall** مدل دارد که در قسمت انتهایی بررسی می شود.

- **unigram(uni_dict, word)** : این تابع احتمال تکی کلمه **word** را با استفاده از دیکشنری محاسبه می نماید. در اینجا از دو مقدار **lambda** 1 و 2 استفاده می شود تا دقت بیشترین شود.

- **is_negative(uni_dicts, bi_dicts, string, mode)** : این تابع با گرفتن دیکشنری های مثبت و منفی و همچنین کلمات پر تکرارشان، بررسی می کند که **string** نظری مثبت یا منفی است.
به این صورت که ابتدا **mode** تعیین می کند که از مدل **unigram** استفاده شود یا **bigram** . سپس رشته را پیش پردازش می کند و کلمات پرتکرار را از آن حذف می کند سپس طبق رابطه گفته شده احتمال را حساب می کند و اگر نسبت احتمال منفی بودن به مثبت بودن از 1.25 بیشتر شد آن را منفی حساب کرده و **True** بر می گرداند.

- **get_string(uni_dicts, bi_dicts)** : این تابع به طور پیوسته از ترمینال رشته نظر دریافت می کند و با توجه به تشخیص مدل دو عبارت **"filter this"** و **"not filter this"** چاپ می کند و با عبارت **!q** متوقف می شود.

- **save_dict(dict, name)** : این تابع دیکشنری را در فایل ذخیره می کند تا بعدا بتوان از آن استفاده کرد.

- **load_dict(name)** : این تابع دیکشنری ذخیره شده را از فایل می خواند و بر می گرداند.

- **test(uni_dicts, bi_dicts, test_dataset)** : این تابع برای تست مدل استفاده می شود. به این صورت که برای یک متن شامل نظر تمام خطوط ها را بررسی می کند و تعداد جملاتی که فیلتر می شوند و نمی شوند را بر می گرداند.

تاثیر λ و ϵ در precision و recall :

اینکه **dataset** ما چقدر جامع و بزرگ باشد تاثیر زیادی در محاسبه احتمال دوتایی و تکی کلمات دارد.

و از آنجایی که **dataset** در این پروژه آنچنان بزرگ نیست درنتیجه احتمالات دو تایی را با دقت کمی بیان می کند.

بنابراین مقادیر λ و ϵ باید طوری باشند که دقت را افزایش دهند و ترکیب احتمالات دوتایی و تکی را با بهترین دقت برگردانند.

پس از بررسی نتایج مختلف، λ و ϵ به صورت زیر مقداردهی شدند.

```
def bigram(uni_dict: defaultdict, bi_dict: defaultdict, word1: str, word2: str):
    m = sum(uni_dict.values())
    max_value = max(uni_dict.values())
    uni_prob = uni_dict[word2] / m
    bi_prob = 0
    if uni_dict[word1]:
        bi_prob = bi_dict[word1 + word2] / uni_dict[word1]
    lambda1 = uni_dict[word1] / max_value / 100
    lambda3 = (1 - uni_dict[word2] / max_value) / 1000
    lambda2 = 1 - lambda1 - lambda3
    epsilon = 1000 / m
    return lambda1 * bi_prob + lambda2 * uni_prob + lambda3 * epsilon


def unigram(uni_dict: defaultdict, word: str):
    m = sum(uni_dict.values())
    max_value = max(uni_dict.values())
    uni_prob = uni_dict[word] / m
    lambda1 = (1 - uni_dict[word] / max_value) / 1000
    lambda2 = 1 - lambda1
    epsilon = 1000 / m
    return lambda1 * epsilon + lambda2 * uni_prob
```

مقادیر به نحوی انتخاب شده اند که اگر احتمال 2 تایی دقت بالایی داشت آن را بیشتر اثر دهند و اگر نه دقت احتمال تکی بیشتر می شود و همینطور اگر دقت احتمال تکی نیز کم بود **lambda3** بیشتر می شود تا تاثیر اپسیلون بیشتر شود.

تاثیر حذف کلمات پرتکرار و کم تکرار در precision و recall :

طبق نتایج گرفته شده از مدل حذف کلمات کم تکرار از بار محاسباتی می کاهد ولی دقت را کاهش می دهد.
همینطور حذف کلمات پرتکرار نیز از بار محاسباتی می کاهد و دقت را کمی افزایش می دهد.

برای آزمایش این دو مورد بررسی شدند و نتایج مطابق زیر است:

```
Negative Precision: 0.81
Negative Recall: 0.80
Negative Score: 0.81
Positive Precision: 0.80
Positive Recall: 0.81
Positive Score: 0.81
```

بدون حذف کلمات پرتکرار

```
Negative Precision: 0.81
Negative Recall: 0.81
Negative Score: 0.81
Positive Precision: 0.81
Positive Recall: 0.81
Positive Score: 0.81
```

با حذف کلمات پرتکرار

همانطور که می بینیم حذف کلمات پرتکرار تاثیر مثبتی در دقت گذاشته است.

- با توجه به پارامتر های ذکر شده بهترین دقت به دست آمد که نتیجه را در بالا می بینیم.
سعی بر آن بوده که **positive score** و **negative score** بیشترین مقدار را به دست آورند.
یعنی برای اینکه مدل خوبی داشته باشیم باید هر دو مقدار **precision** و **recall** را هم برای تعیین مثبت بودن و هم منفی بودن بررسی کنیم تا به میانگین خوبی برسیم.