به نام خدا

تمرین سری اول

درس مبانی پردازش زبان گفتاری

سینا علی‌نژاد

شماره دانشجویی: ۹۹۵۲۱۴۶۹

سوال ۱-

برای این سوال، من مقادیر مختلف min_length, prob_thresh, n_value را امتحان کردم. مقدار min_length را در مواردی که در نوتبوک آوردم همگی ۱۰ است و به نظرم معقول به نظر رسید و مقادیر کوچکتر از آن ممکن بود جملات ناکامل بدهد. برای دو پارامتر دیگر، مقادیر مختلف را تست کردم و نتیجه گیری من به شرح زیر است:

مقدار n_value: وقتی این مقدار کوچک بود (برای مثال۲ یا ۳)، پس از چند کلمه، مدل فراموش میکرد برای مثال داریم درباره inflation (تورم) صحبت میکنیم و از افعالی استفاده میکرد که مناسب این مفهوم نبودند، برای مثال tell، در حالیکه انتظار داریم کلمه ای مثال rise یا decrease یا شبیه آن باشد.

```
# n_value=3, probability_threshold=0.05, min_length=10
# Test the text generator
seed_text = "Inflation is"
generated_text = generate_text(probabilistic_ngram_model, vocab, seed_text, n_value, probability_threshold=0.05, min_length=10)
print(f"Generated Text: {generated_text}")
```

```
Generated Text: Inflation is likely to tell you when youve hit someone
```

اما با بیشتر شدن مقدار n_value این مشکل کمتر میشود. همچنین در n_value بزرگ، تعداد کلیدهایی که وجود ندارند زیاد میشود، در این حالت اگر n-gram وجود نداشت، من از n-1-gram استفاده کردم و اگر آن هم نبود از یک مرتبه پایین تر و در نهایت از unigram استفاده کردم تا این مشکل هم تا حدودی حل بشود.

مقدار prob_thresh:

برای این پارامتر، من سه مقدار 0.01,0.05,0.1 را امتحان کردم و دلیل اینکه از 0.1 بالاتر نرفتم این بود که جملات نصفه و نیمه تولید میشوند به مانند مثال زیر:

```
# n_value=3, probability_threshold=0.1, min_length=10
# Test the text generator
seed_text = "Inflation is"
generated_text = generate_text(probabilistic_ngram_model, vocab, seed_text, n_value, probability_threshold=0.1, min_length=10)
print(f"Generated Text: {generated_text}")
```

```
Generated Text: Inflation is expected to be exchanged for 406 shares of
```

که آخرین کلمه، of است که منطقی نیست.

برای prob_thresh , n های مختلف، خروجی ها در نوتبوک قرار دارد. همچنین برای هر بخش از کد کامنت قرار داده شده است.

سوال ۲-

برای این سوال، از آنجا که naive bayes به ساختار جمله توجهی ندارد و صرفا وجود یا عدم وجود یک سری کلمات را بررسی میکند، در موارد زیادی ممکن است دچار خطا شود.

برای مثال اگر جمله "The movie was not interesting" را به آن بدهیم، ممکن است بخاطر وجود کلمه interesting آن را جزو احساس مثبت دسته بندی کند در حالیکه قبل این کلمه not آمده است که معنا را به کلی عوض میکند. در مثال زیر، باید احساس مثبت پیش‌بینی میشد، ولی منفی پیش‌بینی شد. البته جمله "The movie was not interesting" را به درستی پیش‌بینی کرد. این نشان میدهد که مدل در داده آموزشی، در اکثر مثالهایی که کلمه not بوده، احساس منفی بوده است و کلمه interesting نتوانسته تاثیر خود را بگذارد. اما در جمله عکس زیر، هم کلمه not و هم کلمه bad آمده است و مدل با قطعیت میگوید که احساس منفی دارد.

```
tokens = ["the", "movie", "was", "not", "bad"]
features = get_features(tokens)
predicted_sentiment = classifier.classify(features)
print(predicted_sentiment)
```

```
neg
```

برای حل این مشکل میتوان از bigram ها نیز در تشخیص احساس استفاده کرد.

هرچند که راه حل بهتر استفاده از deep learning و برای مثال یک معماری rnn است.