Comparison and final Report

Naive Bayes and Maxent evaulation metrics results:

Maxent:

test data precision(0.9318181818181818 = (امام test data precision(0.9859154929577465 = (شاه

test data recall(0.9761904761904762 = (امام test data recall(0.9722222222222222 = (شاه

test data F1(0.9534883720930233 = (امام test data F1(0.979020979020979 = (شاه

test data accuracy = 0.9652173913043478 total test data precision = 0.955 total test data recall = 0.974 total test data f1-score = 0.9632

Naive Bayes:

Precision: 0.9043414036956803 Recall: 0.9147347826086956 f1-score: 0.9201540006087957 accuracy: 0.9304347826086956

according to the results you can see that Maxent had a better performance and results rather than NaiveBayes

Naive Bayes

I calculated number of `true_positives` , `true_negatives` , `false_positives` and `false negatives`. Actually I have a dictionary like {'tp': 0, 'tn': 0, 'fp': 0, 'fn': 0} for both of my labels and iterate through my data and calculate it. According to the following formula we can calculate the metrics:

- precision = (tp) / (tp + fp)
- recall = (tp) / (tp + fn)
- f1-score = 2*(precision * recall) / (precision + recall)
- accuracy = (tp + tn) / (tp + tn + fp + fn)

If a prediction is true then we add one to its `tp` and other label's `tn` and the same way for other states .. Example of one sentence estimated wrong in naive bayes:

> امام اصلاحات ارضی آقا به اینجا منتهی شد که یک بازاري درست کرد براي‌آمریکا که آمریکا چیزهایی که باید بریزد دور به ایران بفروشد نفت ما را که دارند این طور می برند بعد از سی سال دیگر)به قول شاه(بعد از سی سال دیگر تمام می شود نفت نه اینکه نفت تمام می شود نخیر تمام دارند می کنندارند با این لوله هاي بزرگ که به اندازه این اطاق شاید بعضی هایشان‌بزرگی اش باشد به اندازه قامت انسان زیادتر بلندیش دورش هست هیکلش هست دارند با زور نفت هاي ما را در می آورند و می فرستند طرف امریکا عوضش هم که باید به ما بدهند عوضش هم اسلحه اي را که می خواهد امریکا بیاید ایران پایگاه داشته باشد در مقابل شوروي باید یک چیزي هم به ایران بدهداگر چنانچه اجازه اش نباید اجازه بدهند لکن حالا این خیانت را کردند و خواستند اجازه بدهند باید یک چیزي هم بدهد به ما که بیاید پایگاه درست بکند نفت ما را می گیرد پایگاه درست می کند براي خودش عوض به ما می دهد یعنی پایگاه درست کردن براي آمریکا یا آن اسلحه هاي بزرگ حتی از ممالک دیگر فرانسه می خرند آن چیزهاي بسیار گران قیمت را که به درد ما نمی خورد به عوضش نفت را دارند می برند طیاره هاي 350میلیون دلاري 350 میلیون

Here our classifier detected this as شاه which is actually امام. The reason is because of Naive Bayes pays attention to `bag-of-words`and the `order` and `sequence` of words is not important to it. And most of the times Shah talked about financial stuffs so it learned that each of these words are more probable in شاه rather than امام. But we should notice that we need a model which can recognize a group of words and it should be like a sequence and not only a unigram but more than that.

Maxent

I defined some features in Maxent : At first compilation I had these features:

1. Length
2. has_arabic
3. unigram(each word of class)

1- It indicates length of sentence because most of the times I saw that Imam Khomeini's speech and usual sentences is much more longer than Shah! 2- I saw that Imam Khomeini uses much more Arabic words rather than Shah. 3- All of us know that each word is a good data to be used as a feature.(except stop words) Then I trained my Maxent modelusing these features Example of model1 sentences given to maxent trainer:

امام خیانتکار:3 و:1 ما:1 این:1 سلطنت:2 نمی:1 خواهیم:1 اصلش:1 از:1اول:1 بوده:1 خوب:1 هایشان:1 هم:1 بد:1 بودند:1 آنهایی:1 که:1 شما:1 یا:1

length:23 has_arabic:0

I evaluated this model metrics and I decided to make it better

Before Optimization Results: (model1)

```
MaxEntTrainer,gaussianPriorVariance=1.0
Summary. test accuracy mean = 0.9565217391304348 stddev = 0.0 stderr = 0.0
Summary. test precision(امام) mean = 0.9302325581395349 stddev = 0.0 stderr = 0.0
Summary. test precision(شاه ) mean = 0.9722222222222222 stddev = 0.0 stderr = 0.0
Summary. test recall(امام) mean = 0.9523809523809523 stddev = 0.0 stderr = 0.0
Summary. test recall(شاه ) mean = 0.9722222222222222 stddev = 0.0 stderr = 0.0
Summary. test f1(امام) mean = 0.9411764705882352 stddev = 0.0 stderr = 0.0
Summary. test f1(شاه ) mean = 0.9722222222222222 stddev = 0.0 stderr = 0.0
```

Then I decides to add bigrams to my feature because I undrestood that the 5 percent problem is almost probelm of sequence of words and it cames from this that unigrams are not enough !

So I added bigrams t my fetures and it became like this:(model2) 1:باشد مستودع 1:شاید زیرا 1:صاف طور 1:همین امام length:7 has_arabic:0 1:مستودعباشد شایدمستودع 1:زیراشاید صافزیرا 1:طورصاف همین‌طور 1:

Then I trained it and I Reached some better results :)) After Optimization Results: (model2)

```
MaxEntTrainer,gaussianPriorVariance=1.0
Summary. test accuracy mean = 0.9652173913043478 stddev = 0.0 stderr = 0.0
Summary. test precision(امام) mean = 0.9318181818181818 stddev = 0.0 stderr = 0.0
Summary. test precision(شاه ) mean = 0.9859154929577465 stddev = 0.0 stderr = 0.0
Summary. test recall(امام) mean = 0.9761904761904762 stddev = 0.0 stderr = 0.0
Summary. test recall(شاه ) mean = 0.9722222222222222 stddev = 0.0 stderr = 0.0
Summary. test f1(امام) mean = 0.9534883720930233 stddev = 0.0 stderr = 0.0
Summary. test f1(شاه ) mean = 0.979020979020979 stddev = 0.0 stderr = 0.0
```

As you can see I could reach accuracy of 96.52 percent which is better than model1.

Wong predicted example: We can refer to wong_predicted.txt and see the wrong ilnes and check them from input.For example the following sentence is predicted امام but it is شاه .

> شاهولت:1 نیز:1 وظیفه:1 دارد:1 که:1 برای:1حل:1مشکلات:1استخدامی:1 و:1 تجدیدنظر:1 در:1 سازمانهای:1دولتی:1 اقدام:1 لازم:1 بنماید:1 length:17 has_arabic:0 1:ولتنیز نیزوظیفه:1 وظیفهدارد:1 داردکه:1کهبرای:1 برایحل:1حلمشکلات:1مشکلاتاستخدامی 1:استخدامیو:1 وتجدیدنظر:1 تجدیدنظردر:1درسازمانهای:1 سازمانهایدولتی:1 دولتیاقدام:1 اقداملازم:1 لازمبنماید:1

As we can see in Test.output.txt:

```
array:20  0.39025361142903714  شاه  0.6097463457307801  امام
```

The reason may be some similar features because as you can see here the probabilities are near(60, 40) and A human can also make mistake in these sentences.

Question of Documnet: So generally i compared these two methods above. (feature having or not having differences and such those things) Totally I observed that Maxent had a better functionality than Naive Bayes But I can't generalize that if a classifier is much more complicated it's better... I can say that if there are much more logical features covered it can have a better complexity And Also we have a bad effect in Naive Bayes which Maxent doesn't have that(Bag-Of-Words Effect) But we also should be careful about over fitting. We should try not to focus on some special features and in that way the model overfits ..