Sentiment Analyser of Sinhala Language using Bigram Language Model

Q1) Sentiment analysis of social network's comments is a common task in the field of text mining. In sentiment analysis predefined sentiment labels, such as "positive" or "negative" are assigned to text sentences. Discuss how bigram model can be used to assign sentiments as "positive" or "negative" to given text sentences. (Assume that you are given a set of positive polarity documents and negative polarity documents.)

The main target of sentiment analysis is the classification of an unknown comment by its sentiment. The approach to classifying the comments by the sentiment can be achieved through applying "Markov Assumptions" to the Bigram language model.

Markov assumptions tell the probability of a word depends only on the probability of a limited history. When comes to bigram language models, the probability of a word depends on the probability of its previous word.

The probability of a sequence of words can be computed by the below equation.

$$P(w_n^1) \approx P(w1, w2, \dots, w_n) \approx \prod_{i=1}^{i} P(W_i \,|W_{i-1}) \qquad (1)$$

To calculate the bigram probability, Maximum Likelihood Estimation (MLE) has been used. MLE computes the bigram probability as follows.

Word sequence= "street address"

$$P(address|street) = \frac{Countc(\text{“}street\ address\text{”})}{Count(\text{“}street\text{”})}$$

$$P(W_i \,|W_{i-1}) = \frac{Count(W_{i-1}W_i)}{Count(W_{i-1})} \qquad (2)$$

But there is a problem in using Maximum Likelihood Estimator, when evaluating the model using the test dataset. Because when there is a new bigram available in the training dataset, it gives the '0' as the bigram probability of the sentence (Because $Count(W_{i-1}W_i)$ will become '0').

Therefore, Laplace Smoothing has been using to better predictions. When applying Laplace smoothing to bigram language models, we consider every bigram had been seen at least once.

$$P^*_{Laplace}(W_i\,|W_{i-1}) = \frac{Count(W_{i-1}W_i)+1}{Count(W_{i-1})+V} \qquad (3)$$

Next, I explain how the above equations are applied practically.

Step 01

- There are two types of comments as positive and negative in the training dataset. Finding bigram token details and unigram token details of the positive and negative comments separately is the main aim of the first step.

- When it comes to social media comments, there can be more than one sentence within a comment. As the first step, each comment needs to be tokenized by sentences.

  Comment:
  "Niyama kalu sinhalaya. Adambarai sahodaya. Thawath chirath kalayak API wenuwen ape ayushath aran jiwath wenna."

  After sentence tokenizing:
  ['Niyama kalu sinhalaya.', 'Adambarai sahodaya.', 'Thawath chirath kalayak API wenuwen ape ayushath aran jiwath wenna.']

- Then each sentence needs to be pre-processed. Pre-processing of the comments before applying them to the model is essential. I will explain the preprocessing of the comments in English and Singlish language models separately.

    - Punctuations removing, numbers removing, removing the starting and ending spaces, stop words removing, converting to lowercase, stemming and lemmatization, adding starting and ending tags are done as the pre-processing in the English language model.

      Sentence: " I liked it 100% "

      1. Removing punctuations
         Output: " I liked it 100 "
      2. Removing numbers
         Output: " I liked it "
      3. Removing starting and ending spaces
         Output: "I liked it"
      4. Removing stop words
         Output: "I liked"
      5. Converting to lower case
         Output: "i liked"
      6. Stemming and Lemmatization
         Output: "I like"
      7. Adding starting and ending tags
         Output: "<s> I like </s>"

    - Only punctuations removing, converting to lowercase, removing the starting and ending spaces, adding starting and ending tags are done as the pre-processing in the Singlish language models. Because in the Singlish language model it is really hard to identify the lemmas and stems without the domain knowledge about the Sinhala language.

Before cleaning the comment:

['Niyama kalu sinhalaya.', 'Adambarai sahodaya.', 'Thawath chirath kalayak API wenuwen ape ayushath aran jiwath wenna.']

After cleaning the comment:

['<s> niyama kalu sinhalaya </s>', '<s> adambarai sahodaya </s>', '<s> thawath chirath kalayak api wenuwen ape ayushath aran jiwath wenna </s>']

- The next step is finding the occurrences of unigram and bigram counts of positive comments list and negative comments list which are in the training corpus separately.

Step 02

- Now we are having the bigram and unigram token frequencies are calculated and stored in a dictionary. Now we have to find the bigram probability of a given new test comment from both positive and negative polarity comments separately.

- The bigram probability of a given test comment needs to be found from both positive and negative corpuses available in the training dataset. Probabilities can be calculated using equation (1) and equation (3).

- By comparing the probability got from the positive and negative corpus, we are able to decide the sentiment is positive or negative. If the probability from the positive corpus is less than the probability from the negative corpus, the sentiment of the test comment is given as ‘Negative’. Otherwise, the sentiment of the test comment is given as ‘Positive’.

Q3) Evaluate the performance of the language model using perplexity

Perplexity is the most commonly used intrinsic evaluation metric. The probability of the test set normalized by the number of words is called 'perplexity'.

$$PP(W) = P(w1, w2, ..wn)^{-1/N}$$

Here, bigram probabilities of all the comments in the test dataset will be calculated and then get the multiplication of those probabilities. Then get the power of -1/N values where N means no of words in the test dataset.

In the below perplexity evaluation, I assume each comment as a unique test dataset. Because when calculating the multiplication of all the bigram probabilities of the comments in the test dataset, it almost equals to value 0.

For the model evaluation, I have calculated the perplexity for each comment in the test dataset. The results as follows.

	comment	label	result	perplexity
0	Obathuma rata athta ththawa dannawa rata unta ...	Positive	negative	4.914088
1	Budu saranai Devipihitai	Positive	positive	7.067377
2	Mama obathumiyata 1994 Indan manapayak denne p...	Positive	positive	211.947881
3	Nithiya hariyata kriyathmaka karanna madam nat...	Positive	positive	169.551981
4	Me dewal hoden karaganayamata obata dahereya s...	Positive	positive	97.920849
5	Madamge kapavima godak watinawa bs	Positive	positive	7.067377
6	Aheta paninawa meheta paninawa.hirikithai oya ...	Negative	positive	109.199277
7	rate minissu corona walin merenawa palakayanta...	Negative	negative	195.420332
8	ledeth haragena chinata rath dunna kalakanni	Negative	positive	7.067377
9	Madamta hamoma garu karanewa ufariema wada kar...	Positive	positive	121.490674
10	Parliament eke kathawak karanna danne nathi ma...	Negative	positive	254.364832
11	Hora-Boru Neti Bandaranayaka Waru	Positive	positive	49.947812
12	Okkoma ekaibaluwama	Negative	positive	18.788294
13	Apoi meyaawa hambenna giyaama meyaata wada ser...	Negative	positive	196.334778
14	Dirgausha labewa	Positive	positive	14.306252
15	Sobahawadarmaya araksha kirimata multhena laba...	Positive	positive	121.490674
16	Ithamath honda sankalpayak mr. President....	Positive	positive	7.067377
17	Godak hodai sir. Jaya wewa.	Positive	positive	10.130346
18	Rata hoda athata yanawa, oya theerana kriyathm...	Positive	positive	109.199277
19	Mekedda ban mu ube boru ahala minissu dan kora...	Negative	positive	143.155938
20	ratema galaum karuwan oba pakshaya pamanai	Positive	positive	81.438737
21	Oba dedenama api garu karana nayakain. Jayawew...	Positive	positive	33.780403

Figure 1: Testing results

As per the observations,

- the model gives lower perplexity values when the test comment looks like the data within the training dataset.

    Ex: - “Budu saranai Devipihitai”

    Unigrams and the bigrams of the above sentence are likely appeared in the training dataset. Therefore, the perplexity is a lower value as per the equation because the probability of the test comment is high.

- the model gives higher perplexity values when the test comment looks different from the data within the training dataset.

    Ex: - “rate minissu corona walin merenawa palakayanta port sity eka chaina walata denna thiyena unanduwa janathawa bera ganna eththmama ne”

    This is a completely new comment to the model. Therefore, the perplexity is a higher value as per the equation because the probability of the test comment is very low.

This is not a good model for industrial usage because this model gives high perplexity value to the comments which consist of data that are not containing in the training dataset.

Q4) Discuss how to improve the classification performance of the above sentiment analyzer.

- Classification performance can be improved by identifying the most commonly used words in the Singlish language. Then the model will able to normalize the effect of the most commonly used words. This can be implemented by introducing stop words to the Singlish language with the help of domain experts.

- Also, the performance of the classification can be achieved by improving the size of the dataset. Then, most of the bigram combinations will be available within the training dataset. It will affect to minimize the perplexity value as well.

- Classification performance also can be improved by using a noise-free dataset. In the comment collection process, lots of noisy comments can be found. As an example, “Ratata aadare minissu” can be considered as a noisy comment. People normally used spellings as ‘adare’ rather than using ‘aadare’. So while selecting the dataset, collecting a noise-free dataset will increase the performance of the classification model.