Data = Phrases from Rotten Tomatoes with sentiment labels from 0-5 where 0 is negative and 5 is positive.

Goal = Label phrases with sentiment from 0 to 5.

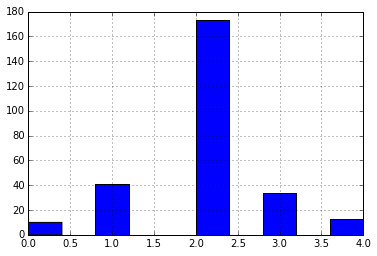
0 - negative

1 - somewhat negative

2 - neutral

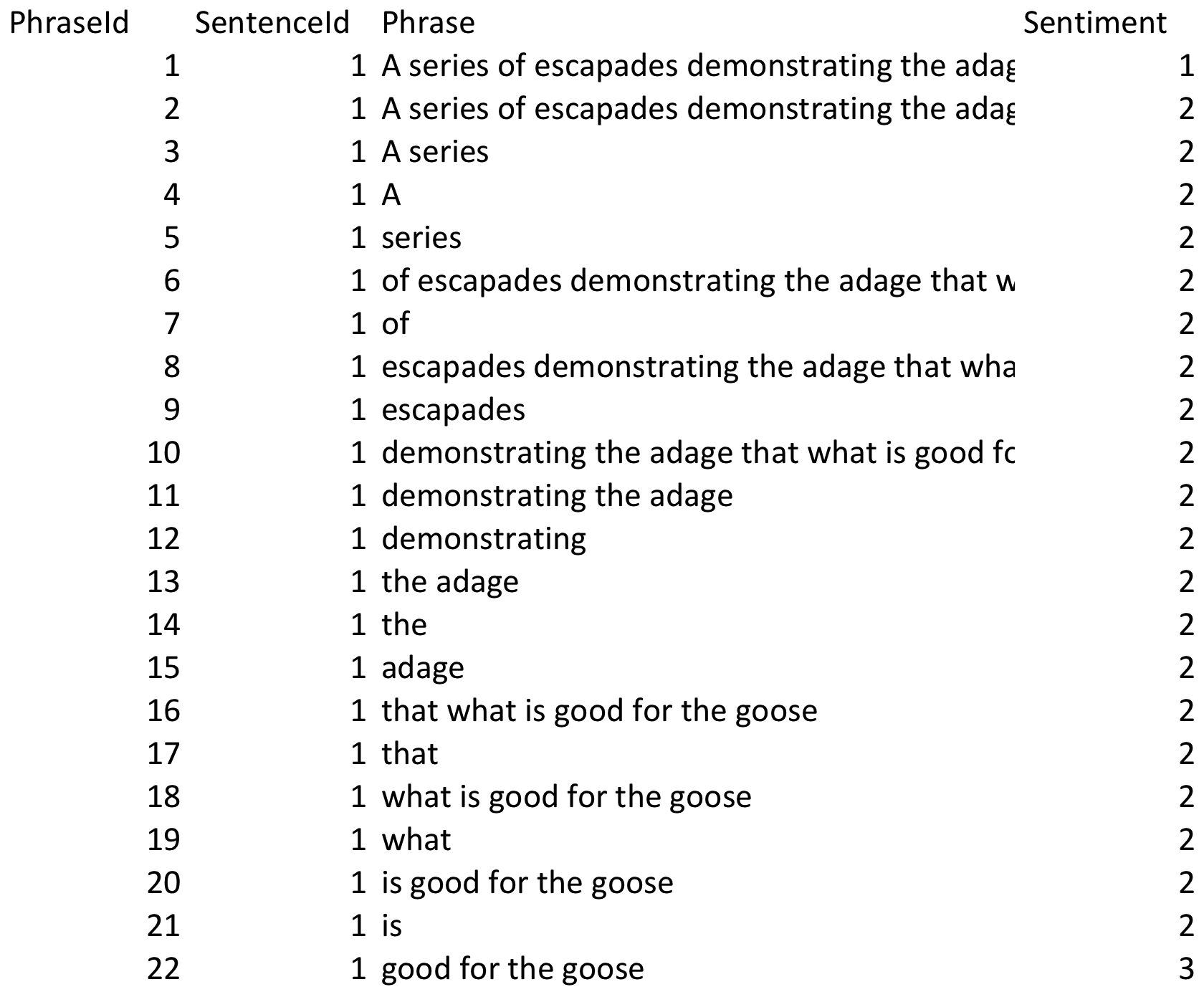
3 - somewhat positive

4 – positive



So far, I have been able to use the Naïve Bayes model like we used in class. Unlike a binary prediction in class, I am predicting a value from 0-5. I have had trouble using all of the data because of “Nan” values appearing. I’ve tried using pandas dropna() function, but it has not worked. Because I got hung up on this, I instead starting working with a small subset of the sample.

With the small sample, I was able to use the CountVectorizer. I’m not sure this is the best approach since the data is both the full phrase and parsed sections of the phrase. For example, the first phrase is contained in the data like this:



Using this approach, I was able to get a count for each of the tokens. For example:

count
token

6
adage

3
aggressive

6
all

4
also

11
amounts

7
amuses

23
and

11
bartlett

3
being

9
betrayal

Once I predict the test data, I only get an accuracy of 62%. I’m wondering if using the full sample will show similar results. Confusion matrix shows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** |
| **1** | 0 | 4 | 0 | 0 | 0 |
| **2** | 2 | 5 | 2 | 0 | 0 |
| **3** | 0 | 7 | 33 | 2 | 1 |
| **4** | 0 | 0 | 3 | 4 | 0 |
| **5** | 0 | 0 | 0 | 5 | 0 |

Because it is not binary, I am not able to do an ROC/AUC test. Is there another test?

Since this project was a kaggle competition, the forum shows that some people were successful using a random forest. Once we learn this method, I would like to see how well this can be applied.

Key next steps include:

* Understanding how to deal with “Nan” values
* Apply another measure/metric for testing the fits that can be used for categorical predictions
* Understand if I should manipulate the data to not use each phrase AND the parsed sections
* Use random forest method

I’d like to do something more interesting from there if time permits. Open to suggestions.