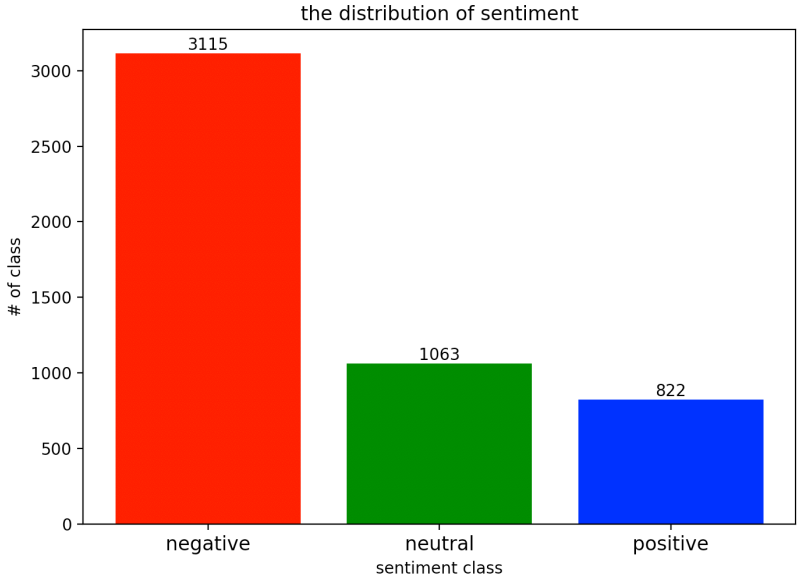
Sentiment Classification Report

Zid: z5235878

Name: Zhaokun Su

1. There are three class labels for 5000 tweets in total: negative (3115), neutral (1063) and positive (822), and we could see the distribution below figure:



2. For our classification problem, it can be shown that each instance is contained in one class and be predicted to be in one class, the value of micro-precision, micro-recall and micro-F1 is the same according to the definition of three metrics. We get micro-F1 = micro-precision = micro-recall = accuracy

Model	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy
BNB (all vocabulary)	**0.83**	**0.52**	**0.57**	**0.73**
BNB (top 1000 frequence)	**0.72**	**0.74**	**0.73**	**0.79**

Model	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy
MNB (all vocabulary)	**0.80**	**0.59**	**0.64**	**0.76**
MNB (top 1000 frequence)	**0.73**	**0.71**	**0.71**	**0.79**

We could see a result that if we use features of the most frequent 1000 words from vocabulary to train our model, all metrics improve except macro-precision value which goes through a decline. Classification accuracy, on the whole view, increase to both 0.79 after predicting.

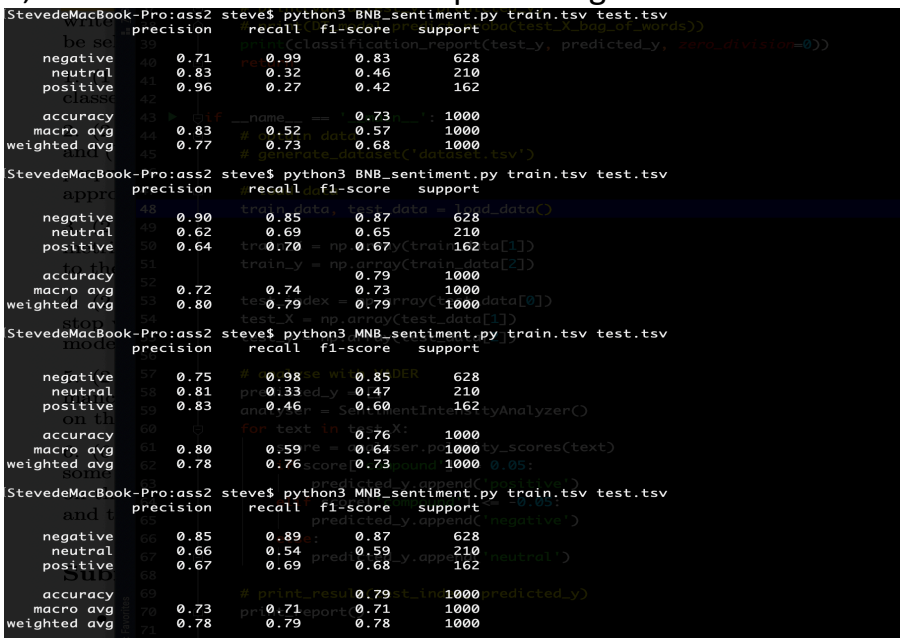


Figure above is the correspongding classification report for mentioned two models.

3. First we need to use Vader to analyse our test sentences, interestingly, this process does not need to be pre-processed by our filters.
We can directly use data from test sentences. Finally, we will generate our baseline based on Vader analyzer.

Model	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy
Vader	**0.54**	**0.60**	**0.51**	**0.54**
BNB (all vocabulary)	**0.83**	**0.52**	**0.57**	**0.73**
BNB (top 1000 frequence)	**0.72**	**0.74**	**0.73**	**0.79**
MNB (all vocabulary)	**0.80**	**0.59**	**0.64**	**0.76**
MNB (top 1000 frequence)	**0.73**	**0.71**	**0.71**	**0.79**
Decision Tree (all vocabulary)	**0.64**	**0.58**	**0.59**	**0.71**
Decision Tree (top 1000 frequence)	**0.64**	**0.58**	**0.59**	**0.71**

As we can see, our Vader analyzer requires a relative lower accuracy in terms of this classification problem, and every indicator/metrics value is not high, which gives us a bad performance in this issue, and why this happen?
The reason why the accuracy of VADER is difficult to anticipate is mainly because:
(i) crowd- sourcing is in general highly unreliable, and
(ii) this dataset might not include much use of emojis and other markers of sentiment.
In terms of other models, they provide better performances than baseline, and it seems like using top 1000 frequent words surprises us the most no matter what model we use, and we could conclude for this problem with respect to performance:
BernoulliNB Model and MultinomialNB Model perform better than Decision Tree model in this specific problem.

4. In order to make a clear comparasion, we establish a table before and after removing stopwords and applying stem words:
Operations: means we do operation a (remove stopwords) and operation b (apply stem words);

Model	Before operations				After operations			
	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy
BNB (top 1000 frequence)	0.72	0.74	0.73	0.79	0.72	0.72	0.72	0.78
MNB (top 1000 frequence)	0.73	0.71	0.71	0.79	0.71	0.69	0.70	0.77
Decision Tree (top 1000 frequence)	0.64	0.58	0.59	0.71	0.64	0.59	0.61	0.70

From our experiments, we could see it seems to get a slightly change on different macro-metrics after removing stop words and applying stem words. Interenstingly, the micro-indicator (accuracy) decreases a very very little for all of our models, in turn potentially, we can say our models seems like finding it hard to get an improvement when we apply stopwords and stem words approaches to pre-process sentences.

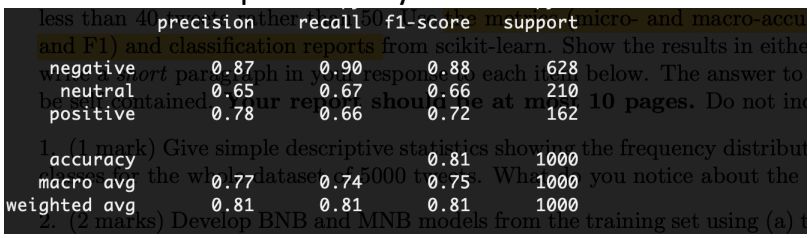
5. we change our lower_case == False, which means we use our data set without conversion to lower case on the test set:

Model	With lower case conversion				Without lower case conversion			
	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy
BNB (top 1000 frequence)	0.72	0.74	0.73	0.79	0.72	0.74	0.73	0.79
MNB (top 1000 frequence)	0.73	0.71	0.71	0.79	0.71	0.69	0.70	0.77
Decision Tree (top 1000 frequence)	0.64	0.58	0.59	0.71	0.62	0.54	0.56	0.70

From table, we could know that the performance with lower case conversion and without lower case conversion are almost the same, after setting lower_case == False, our metrics change little. As we can see, accuracy metrics for MNB model and DT model are decreased slightly.
In conclusion, making a conversion to lower case or not seems do not influence our model accuracy a lot, if we can, choosing another aspect to improve our model performance should be worthy.

6. why choose this model?
After a serious of comparasion, it turns out BNB model surprise us a lot. We establish our best model by choosing BNB basic model and tuning hyper parameters:
For pre-processing data and models parameters::
(1) we do remove all urls;
(2) we do remove junk characters;
(3) we **do not** remove stop words (during experiments before, we know stop words affect performance little);
(4) we do use stem words (during experiments before, we know only using words makes BNB performance a little better);
(5) we choose the most 1400 frequent words and use conversion to lower case (influence performance a lot);
(6) we choose BernoulliNB model with alpha = 2.0 (1.0 by default)
Experimental analysis of my own model?

The figure of classification report for my own best model is shown below:



	precision	recall	f1-score	support
negative	0.87	0.90	0.88	628
neutral	0.65	0.67	0.66	210
positive	0.78	0.66	0.72	162
accuracy			0.81	1000
macro avg	0.77	0.74	0.75	1000
weighted avg	0.81	0.81	0.81	1000

We know from theory course: If the number of different classes is the same, average values of macro and micro do not differ so much. But if they differ a lot, we use macro values to evaluate large sample class if we focus more on large sample class. And using micro values to evaluate small sample class if we focus more on small sample class. My model performs the best on both of them, which indicates we might do the right thing.

d. table of different models final comparasion:

Model	Macro-precision	Macro-recall	Macro-f1	Micro-precision/ Micro-recall/ Micro-f1/ Accuracy
Vader	*0.54*	*0.60*	*0.51*	*0.54*
BNB (all vocabulary)	0.83	0.52	0.57	0.73
BNB (max_features = 1000)	0.72	0.74	0.73	0.79
MNB (all vocabulary)	0.80	0.59	0.64	0.76
MNB (max_features = 1000)	0.73	0.71	0.71	0.79
Decision Tree (all vocabulary)	0.64	0.58	0.59	0.71
Decision Tree (max_features = 1000)	0.64	0.58	0.59	0.71
BNB (best method) (alpha = 2.0; max_features = 1400; Apply stem words)	*0.77 (high value)*	*0.74 (best value)*	*0.75 (best value)*	*0.81 (best value)*

Although our model gets a relatively low macro-precision value compared with BNB model (all vocabulary) and MNB model (all vocabulary), the other metric perfomances are all the best, which gives us a satisified result.