COMP 6721 Project 2 Report

Chenwei Song

40024460

Introduction and technical details

In this project, we refer to information retrieval, term-frequency computing, Naïve Bayes classifiers and four evaluation methodology such that Accuracy, Recall, Precision, F-measure, respectively.

In information retrieval, term-frequency-inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modelling.

In machine learning, Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong(naive) independence assumptions between the features. They are among the simplest Bayesian network models.

Accuracy:

% of instances of the test set the algorithm correctly classifie, when all classes are equally important and represented. It means that % of instances of the test set the algorithm correctly classifies

Recall, Precision & F-measure:

When one class is more important than the others

Programming language: python3.6

Programming tool: PyCharm

Programming environment: Windows 10

Experiment dataset:

Dataset: hn2018_2019.csv

Training dataset: the CSV file contains in Title which is Created At 2018

Testing dataset: the CSV file contains in Title which is Created At 2019

Four types of class: story, ask_hn, show_hn, and poll

Pre-Processing dataset

We execute several filtering operations to pre-process data, as follow:

Step1: replacing non-ascii to space

Step2: fold the Title to lowercase

Step3: extract special character '[~@$*+_-$&-#{}]'

Step4: extract corpus by bi-gram

In order to use the bi-gram efficiently, I create a bi-gram file named '9.csv'.

This file summaries the corpus without non-ascii code, stop-words, punctuation, number, and date. All of them have high frequency with a pattern '<JJ>*<NN>'. 'JJ' is adjective, 'NN' is noun.

For each title, if it contains corpus that exist in bi-gram file, extract it as a corpus.

Step4: word lemmatization

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization is similar to stemming but it brings context to the words, So it links words with similar meaning to one word.

Text pre-processing includes both Stemming as well as Lemmatization. Many people find these two terms confusing. Some treat these two as same. Actually, lemmatization is preferred over Stemming because lemmatization does morphological analysis of the words.

Examples of lemmatization

-> rocks: rock

-> corpora: corpus

-> token: take

After those steps, we get vocabulary.txt.

If the word contains special punctuation or it is not noun, verb, adjective, or adverb, we save it in remove_word.txt.

1. Experiment 1

1.1 baseline experiment

In experiment #1, we will train our model for baseline experiment on the training dataset. First, we compute the prior for each class and conditional probability for each word or corpus in the vocabulary. For each word or corpus, computer their frequencies and conditional probabilities, the smooth factor $\delta = 0.5$. Then, we use their frequencies and the probabilities of each Post Type class(story, ask_hn, show_hn, and poll) to compute score and then predict the Post type for testing dataset.

1.2 Results and analysis

By analysing the result, we evaluate the baseline experiment with the test set by these values: Accuracy, Precision, Recall, $F_1 - measure$.

Table 1. Analysis of baseline experiment

Table 1. Analysis of baseline experiment $\delta = 0.5$

Class / Evaluation	Story	Ask_hn	Show_hn	Poll
Precision (P)	125511 / 135207 = 92.82%	167 / 301 = 55.48%	103/214 = 48.13%	5/1493 = 0.33%
Recall (R)	125511 / 126852 = 98.94%	167 / 5454 = 3.06%	103/4903=2.1%	5/6 = 83.33%
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.78%	5.80%	4.02%	0.00%
Accuracy of Model	91.67%			

Table 2. Confusion matrix of baseline experiment

		Actual class				
		Story	Ask_hn	Show_hn	poll	Total
Classes assigned by our model	Story	125511	4943	4750	1	135207
	Ask_hn	121	167	13	0	301
	Show_hn	103	8	103	0	214
	Poll	1116	334	36	5	1493
	Total	126851	5452	4902	6	137209

For the baseline experiment, the result shows that the model has high Precision and Recall for Post Type 'Story'. For Post Type 'Ask_hn' and 'Show_hn', the model has high Precision, while the evaluation of Recall is much lower. In contrast, Recall of 'Poll' is much high, and Precision of that much low. F1-measure for 'Ask_hn', 'Show_hn', 'Poll' is almost 0%.

Depending on table 1 and 2, we guess the reason that Precision and Recall and F1-measure for the three classes 'Ask_hn', 'Show_hn', and 'Poll' with low value is the training data of those three classes much less than 'Story'. It means the frequency of most of words in 'Story' is much higher than others.

The confusion matrix shows that the baseline classifier makes more errors for 'poll' and does well in 'Story'.

2. Experiment 2

2.1 Stop-word filtering

In the experiment of stop-word filtering, we remove the given stop words from the vocabulary and ignore the stop words in Titles, because stop words are the words that show up a lot in documents, such as prepositions, pronouns, etc.[1], and it does not help a lot for classification.

First, we compute the conditional probability for each word and use the same smoothing factor ($\delta = 0.5$) and the same prior as in experiment #1. And then, we use the prior and the updated likelihood to calculate the scores and label the 'Title' on the test set.

2.2 Results and analysis

After removing the stop words in vocabulary and experimenting with the classifier, we show the test results in the same way as in experiment #1.

Table 3. Analysis of stop-word filtering experiment

Class / Evaluation	Story	Ask_hn	Show_hn	Poll
Precision (P)	126006 / 135924 = 92.7%	155 / 301 = 53%	112/214 = 42.9%	4/1493 = 0.5%
Recall (R)	126006 / 126852 = 99.3%	155 / 5454 = 2.8%	112/4903=2.2%	4/6 = 66.67%
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.90%	5.39%	4.33%	0.01%
Accuracy of Model	92.02%			

Table 4. Confusion matrix of stop-word filtering experiment

		Actual class				
		Story	Ask_hn	Show_hn	poll	Total
Classes assigned by our model	Story	126006	5165	4751	2	135924
	Ask_hn	97	155	40	0	292
	Show_hn	15	134	112	0	261
	Poll	734	0	0	4	738
	Total	126852	5454	4903	6	137209

The result tables show that the model has a higher recall for story and higher precision for story. It means that in stop-word filtering, the model labels well in all titles that are supposed to be story, but it also has a high error rate in other types' identification. The accuracy of 'Story' and 'Show_hn' increases, while the accuracy of 'Show_hn' decreases, and that of 'poll' changes slightly.

And the stop-word filtering has the accuracy of 92.02% overall which is slightly higher than the baseline experiment.

3. Experiment 3

3.1 Word-length filtering

In the word-length filtering, instead of removing the given stop words, we will remove all the words which length is equal or shorter than 2 and words which length is equal or longer than 9 from the vocabulary. And then compute the conditional probability for the rest of the words, and we use the same smoothing factor ($\delta = 0.5$), and prior is also the same as in the previous experiments.

3.2 Results and analysis

After we classify on the title with the test set, here are the same 2 tables which show the results and performance:

Table 5. Analysis of word-length filtering experiment

Class / Evaluation	Story	Ask_hn	Show_hn	Poll
Precision (P)	125341 / 135103 = 92.7%	183 / 409 = 44%	204/595 = 34.2%	3/1108 = 0.2%
Recall (R)	125341 / 126852 = 98.8%	183 / 5454 = 3.3%	204/4903=4.1%	3/6 = 50%
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.69%	6.2%	7.33%	0.5%
Accuracy of Model	91.63%			

Table 6. Confusion matrix of word-length filtering experiment

		Actual class				
		Story	Ask_hn	Show_hn	poll	Total
Classes assigned by our model	Story	125341	5118	4640	3	135103
	Ask_hn	206	183	20	0	409
	Show_hn	373	17	204	0	595
	Poll	931	134	38	3	1108
	Total	126851	5452	4902	6	137209

As we can see in the first table, the overall accuracy of word-length filtering is 91.63%, as same as in experiment #1 and #2, it does better in 'show_hn' and 'ask_hn' classification because the model has 1% higher F_1-measure performance for 'show_hn' and 'ask_hn' than others considering both precision and recall.

In the confusion matrix, it shows that the model misses more story titles than other types of title, but the model has lower error rate for labelling 'show_hn' and 'ask_hn'.

4. Experiment #1, #2, #3 comparison

In this section, we will analyze the results (Accuracy, Precision, Recall, F_1-measure) given by the baseline model, stop-word filtering and word-length filtering. We will show the confusion matrix of 3 experiments in the same table and the performance change table, and then we will compare the performance of the three models.

Table 7 & 8. Analysis of experiment #1, #2, and #3

Changes in experiment #2 and #3 compared with experiment #1

Experiment \ Evaluation		Accuracy	Recall			
		model	story	Ask_hn	Show_hn	poll
#1	Baseline	91.67%	98.94%	3.06%	2.1%	83%
#2	Stop-word	92.02%↑	99.33%↑	2.8%↓	2.2%↑	66.67%↓
#3	Word-length	91.63%↓	98.8%↓	3.35%↑	4.1%↑	0.5%↓

Experiment \ Evaluation		Accuracy	Precision			
		model	story	Ask_hn	Show_hn	poll
#1	Baseline	91.67%	92.82%	55.4%	48.1%	0.33%
#2	Stop-word	92.02%↑	92.7%↓	53.08%↓	42.91%↓	0.54%↑

#3	Word-length	91.63%↓	92.7%	44.7%↓	34.28%↓	0.27%↓
Experiment \ Evaluation		Accuracy	F1-measure			
		model	story	Ask_hn	Show_hn	poll
#1	Baseline	91.67%	95.7%	5.80%	4.02%	0.66%
#2	Stop-word	92.02%↑	95.9%↑	5.39%↓	4.33%↑	1.07%↑
#3	Word-length	91.63%↓	95.6%↓	6.24%↑	7.42%↑	0.53%↓

Comparing the 3 experiments, stop-word model and word-length model do not both increase the accuracy of the model, however, the stop-word model performs greater improvements than the word-length model. That means that short-word and long-word removal achieves a worse result than stop-word removal.

Furthermore, the stop-word model has highest value of Recall for 'poll' than others. We can see the changes of value for 'poll' is larger than others. The evaluation value of other Post types changes slightly.

Table 9. Confusion matrix of the first three experiments

		Actual class					
		Experiment	Story	Ask_hn	Show_hn	poll	Total
Classes assigned by our model	Story	Baseline	125511	4943	4750	1	135205
		Stop-word	126006	5165	4751	2	131324
		Word-length	125341	5118	4640	3	135102
	Ask_hn	Baseline	121	167	13	0	301
		Stop-word	97	155	40	0	292
		Word-length	206	183	20	0	109
	Show_hn	Baseline	103	8	103	0	214
		Stop-word	15	134	112	0	261
		Word-length	373	17	204	0	594
	poll	Baseline	1116	334	36	5	1491
		Stop-word	734	0	0	4	738
		Word-length	931	134	38	3	1106

We can also analyse on the confusion matrix table. All of the models in 3 experiments make more mistakes in Title identification than ask_hn, show_hn, poll_hn, but stop-word model does the least mistakes because it removes the words that are useless and meaningless. Word-length model may remove some words that can identify than ask_hn, show_hn, poll_hn efficiently, so the result is not better than others.

5. Encountered difficulties and interest on experiment #1, #2, #3

5.1 Difficulty and improvement

Difficulty. Independent assumption
Naïve Bayes classifier model assumes that the attributes are independent with each other. However, this assumption is not always applicable in practical applications. When the number of attributes is large or the correlation between attributes is tight, the efficiency of the classifier model might be inferior to the decision tree model.

To mitigate this disadvantage, we come out with possible improvement ideas as the following:

Improvement ideas: cluster the associated keywords
The above experiments are based on the assumption where each word occurs independently. In fact, there is a certain correlation between the occurrence of keywords. If the words with higher correlations are clustered at the beginning, then the simplicity may be used for these related words, and Bayesian model results will be more reasonable.

5.2 Interests and inspirations

Interest. While we were searching for materials online, we found various methods for improving the basic model, such as fancy smoothing strategies, using n-grams, etc. But one that interested us was spacy and textblob. These two packages can recognize words more smartly, we can classify words more efficiently. Moreover, we found that sklearn is a very useful tool that can process data really fast.

Inspiration. If we can filter the words more efficiently, the result will be better. The word length does not effect the result largely.

6. Experiment 4

6.1 Infrequent word filtering

Removing infrequent words may result in better performance because keywords which occur in lesser frequency in the corpus usually does not play an important role in classification. So by removing those unimportant words, we can increase the frequency of important keywords in order to result in higher accuracy.

While removing the top frequent words might lead to lower accuracy because we guess that higher frequent words are most likely the keywords which are discriminating words, and these words play a significant role in computing scores. But we will check the results and verify our idea.

In experiment 4, we will show infrequent word filtering. Firstly, we use a dictionary to store all the words in the training set. Next, we compute frequency for all words, and we gradually remove the words with a certain number of frequency or the words with a certain proportion from the topmost frequent words from the vocabulary. Then, we compute the conditional probability of each word in vocabulary using the same smoothing factor ($\delta = 0.5$). Finally, we use the computed probabilities and the same prior to calculating the score(story) and score(ask_hn) score(show_hn) score(poll) to classify the titles in the testing set.

6.2 Results and analysis

In this section, we will show a table that represent the performance of the classifier. And we will compare the different word removal strategies and discuss by removing how much proportion based on the word frequency, the performance achieves the best. In the end, we will plot the performance of the classifier against the number of words left in the vocabulary.

Table 10. Analysis of infrequent word filtering experiment

Class / Evaluation	Story%	Ask_hn%	Show_hn%	Poll%
Removing words with frequency = 1				
Precision (P)	93.19	52.92	37.51	0.21
Recall (R)	97.91	8.28	6.03	3.33
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.49	14.33	10.4	0.43
Accuracy of Model	91.06			
Removing words with frequency ≤ 5				
Precision (P)	93.64	50	32.34	0.14
Recall (R)	95.96	13.56	11.15	1
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	94.78	21.34	16.59	0.29
Accuracy of Model	89.66			
Removing words with frequency ≤ 10				
Precision (P)	93.76	48.34	31.65	0.11
Recall (R)	94.97	14.44	12.46	1
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	94.36	22.24	17.88	0.23
Accuracy of Model	88.82			
Removing words with frequency ≤ 15				
Precision (P)	93.81	47.30	31.49	0.10
Recall (R)	94.39	14.50	12.93	1
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	94.10	22.20	18.33	0.20
Accuracy of Model	88.31			
Removing words with frequency ≤ 20				
Precision (P)	93.85	46.53	31.43	0.09
Recall (R)	93.97	14.39	13.19	1
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	93.91	21.98	18.58	0.18
Accuracy of Model	87.92			
Removing words with the top 1%				
Precision (P)	92.57	23.92	26.96	0
Recall (R)	99.58	1.11	2.59	0
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.94	2.13	4.72	0
Accuracy of Model	92.19			
Removing words with the top 5%				
Precision (P)	92.54	14.72	22.18	0
Recall (R)	99.5	0.69	2.48	0
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.90	1.33	4.47	0
Accuracy of Model	92.11			
Removing words with the top 10%				
Precision (P)	92.53	13.01	21.37	0
Recall (R)	99.49	0.64	2.40	0
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.85	1.22	4.32	0
Accuracy of Model	92.09			
Removing words with the top 15%				
Precision (P)	92.52	12.15	20.34	0

Recall (R)	99.52	0.56	2.16	0
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.89	1.08	3.90	0
Accuracy of Model	92.10			
	Removing words with the top 20%			
Precision (P)	92.52	12.5	19.56	0
Recall (R)	99.53	0.55	2.01	0
F_1-measure ($\beta = 1$) $(\beta^2+1)PR/(\beta^2P+R)$	95.90	1.05	3.66	0
Accuracy of Model	92.11			

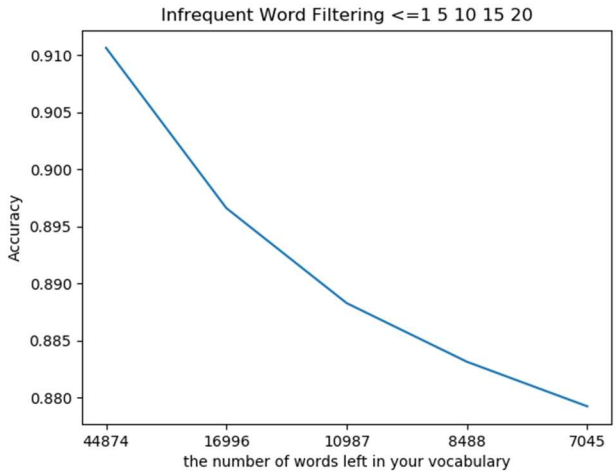


Figure1 accuracy of removing infrequent word with frequency less than [1,5,10,15,20]

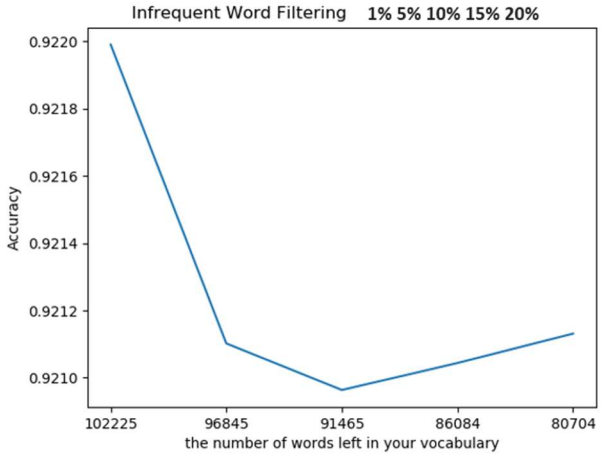


Figure2 accuracy of removing infrequent word with top frequency% [1,5,10,15, 20]

As we can see in figure 1 and 2 the accuracy decrease significantly with the number of words in vocabulary decreasing. While in figure 2 , the accuracy has a temporary slightly increase. Moreover,

the accuracy in figure2 is always higher than that in figure1. We can get a conclusion that removing topmost frequency% is better than remove words with low frequency. However, the fluctuations for figure 1 and 2 are small, just around 2%, even 0.1%.

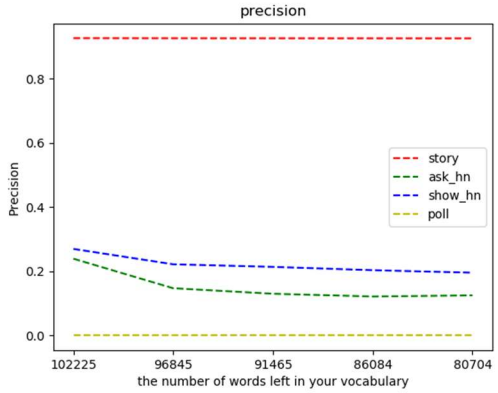


Figure3 Precision of removing infrequent word with top frequency % [1,5,10,15, 20]

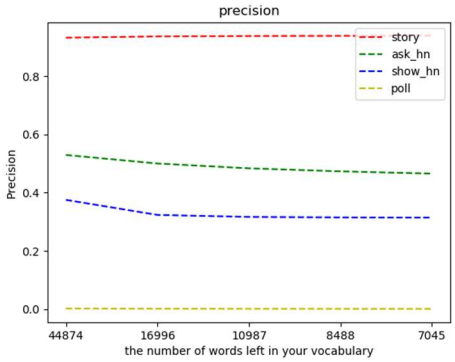


Figure4 precision of removing infrequent word with frequency less than [1,5,10,15,20]

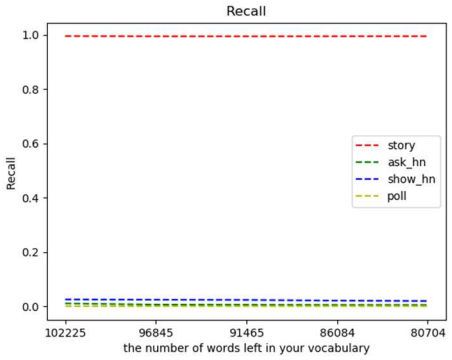


Figure5 recall of removing infrequent word with frequency less than [1,5,10,15,20]

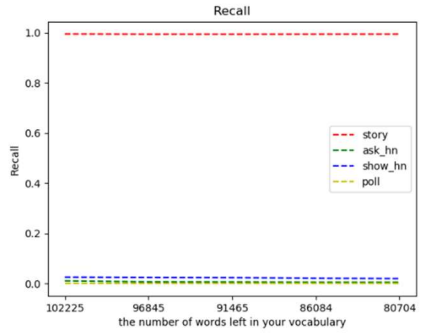


Figure3 Recall of removing infrequent word with top frequency % [1,5,10,15, 20]

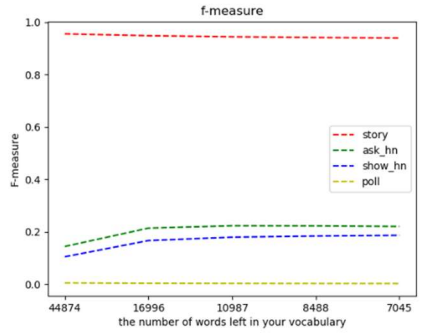


Figure7 f-meaure of removing infrequent word with frequency less than [1,5,10,15,20]

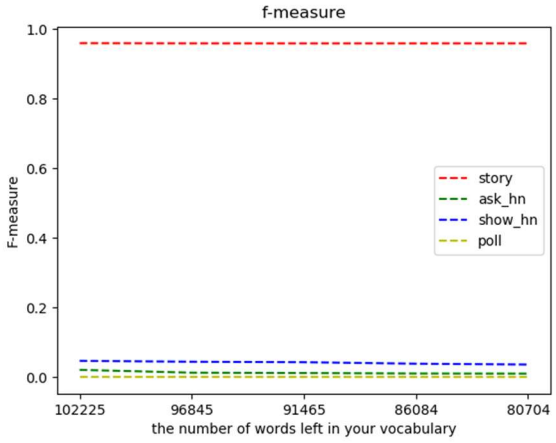


Figure8 f-measure of removing infrequent word with top frequency % [1,5,10,15, 20]

We can see that removing word does not have significant effect for the value of recall, precision, f-measure. We guess the reason is the number of Title labelled by ‘story’ are much large. It means

that the score(story) is always higher than others. If we want to improve the result, the size of training dataset for four classes should keep balanced.

7. Experiment 5

7.1 Experiment 5: change smoothing

As we have known, for the reason that a word, especially person names, proper noun, etc., which does not appear in any training text but appears in the test set leads to a score of 0. To balance this, smoothing plays an important role in Naïve Bayes classification. Commonly, additive smoothing is usual a component in the model. Most cases, we use add-one smoothing, but in practice, a smaller value is preferred.

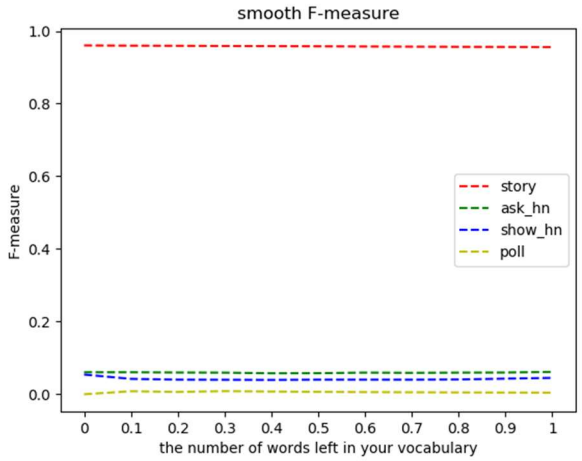
The goal of smoothing is to increase the zero-probability words to a small probability in order to avoid a never-seen word will lead to the entire process failing. And it not only improves the accuracy of the language model but also accommodates the generation of new words and non-informative words.

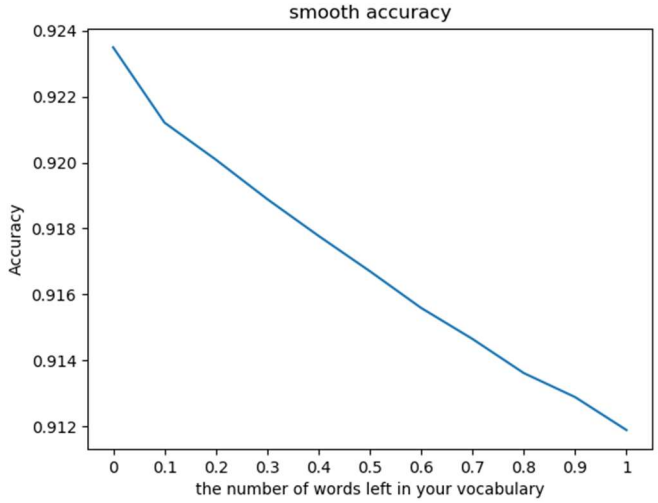
In this experiment, we will do different smoothing factor trials from no smoothing to $\delta = 1$ by increasing 0.1 at each step. For each step, we calculate the conditional probability for each word in the vocabulary and use them to compute the scores. And in the end, we will plot the performance of the classifier against the smoothing value.

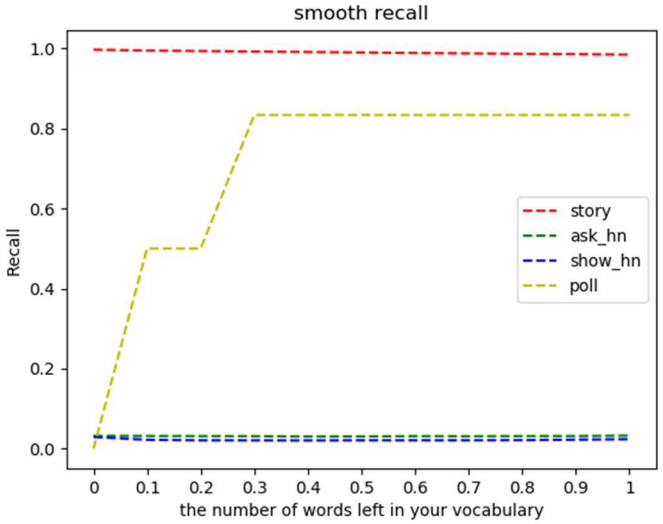
7.2 Results and analysis

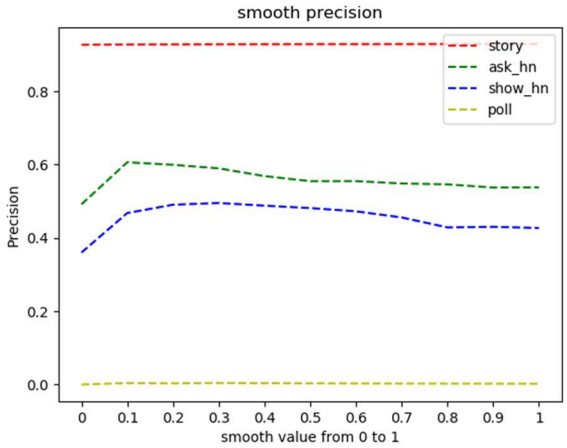
In this section, we will analyse the results and evaluate which smoothing factor suits the best for the model.

We can see that if the dataset is really large(such as more than 4,00,000 data). The value of smooth does not change the result significantly. The largest gap of accuracy is just 1%. We are curious that if we use small training dataset(such that less than 1000 data), what the results of accuracy are. No large changing for the F-measure, precision, and accuracy. However, the value of recall for 'poll' increases obviously. It is because the number of right identify for 'poll' increases.




smooth accuracy
Accuracy
0.924
0.922
0.920
0.918
0.916
0.914
0.912
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
the number of words left in your vocabulary


smooth recall
Recall
1.0
0.8
0.6
0.4
0.2
0.0
story
ask_hn
show_hn
poll
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
the number of words left in your vocabulary


smooth precision
Precision
0.8
0.6
0.4
0.2
0.0
story
ask_hn
show_hn
poll
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
smooth value from 0 to 1

8. Experiment #2, #3, #4 #5 comparison

In this section, we will find out the correlations between the 4 experiments. And we compare the corresponding experiments and the results.

First, we found that the experiment #2 stop-word filtering has some relationship with experiment #4 infrequent word filtering to some extent. Because stop words such as language stop words (e.g. “a”, “the”, “on” etc.) are the most commonly used words in a corpus [4]. And as we discussed in experiment #4, when we remove the top 5% most frequent words in the vocabulary, it did help the model to be more accurate. We will draw a graph that shows the accuracy of the model in the baseline model, stop-word filtering and infrequent word filtering:

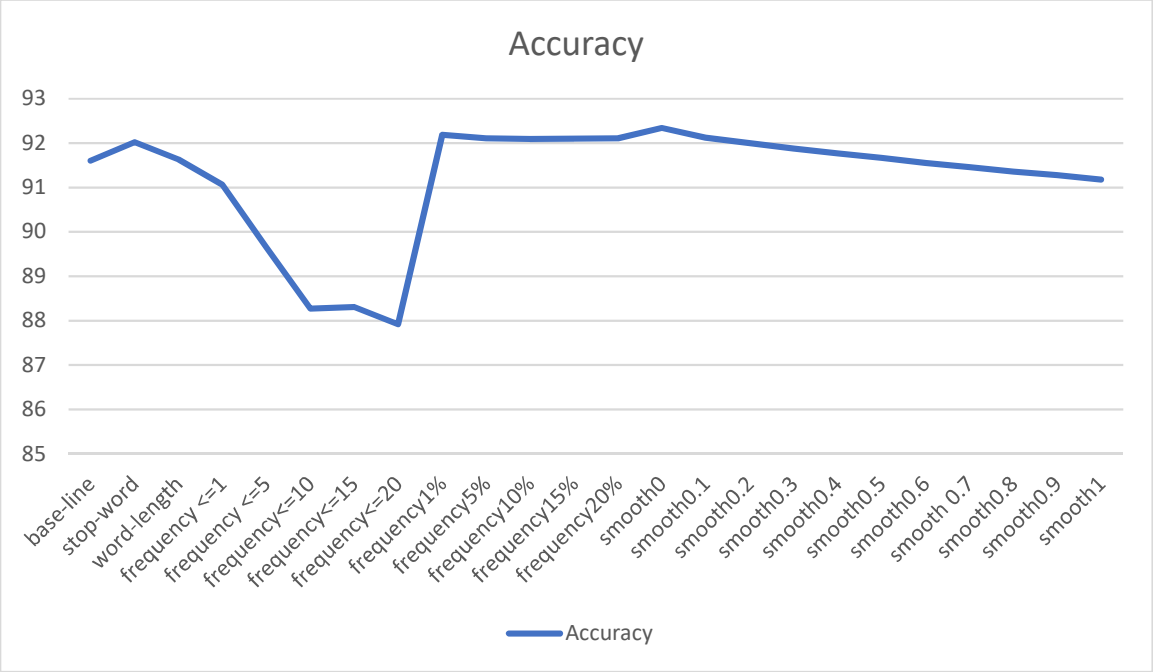


Figure 9 Accuracy for experiment 1# 2# 3# 4#

We found that the experiment #2 stop-word filtering has some relationship with experiment #4 infrequent word filtering to some extent.

We can clearly see that the most efficient filtering method is stop-word and removing word frequency %1. After checking the vocabulary, we found that stop-word looks like same with removing word frequency %1. For example, the removed words are can, be, how, when, what, why…

Those words are neither noun or verb, adj, adv. We can conclude that those words are no contribution to identify the type of title.

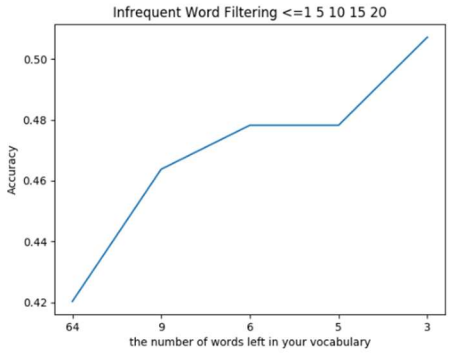
The lowest accuracy is around frequency [5,20]. When we check the vocabulary, we found that those removing words can increase the score of right Post type. Although they are not most frequent, they have important meaning.

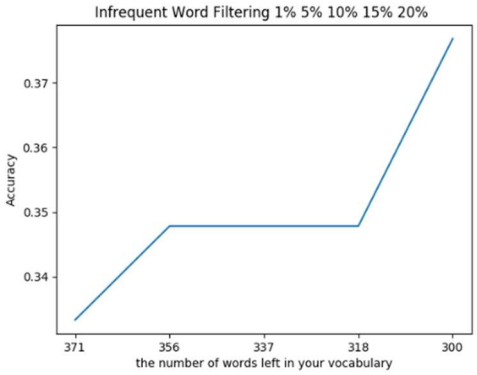
Second decreasing happened with the increasing of smooth. We found that the accuracy will reduce gradually with the increasing of smooth (from 0 to 1).

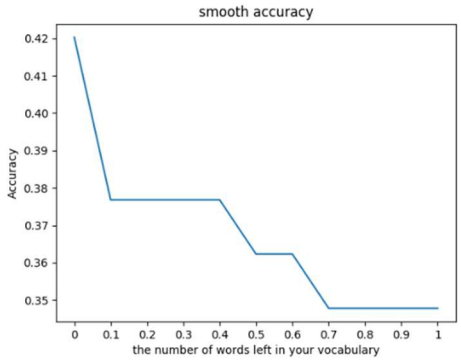
9. Interesting and investigation

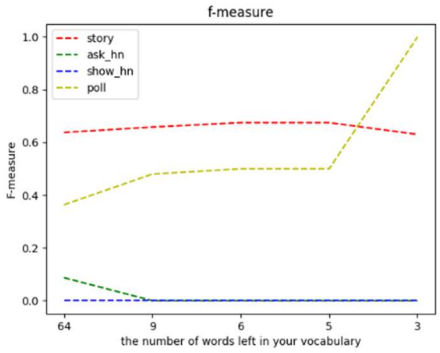
We are curious when the number of each Post type is same, what the performance for Experiment1#, 2#, 3#, 4#, 5#

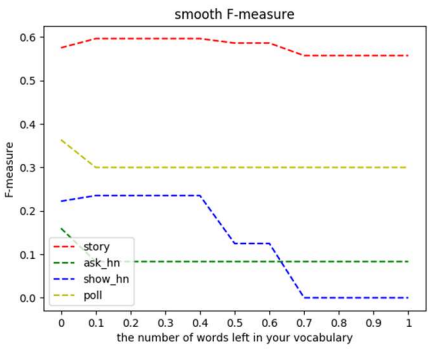
Therefore, we design a small dataset named 100test.csv to do the experiment. The result of Recall, Precision, F1-measure, and accuracy as follow:

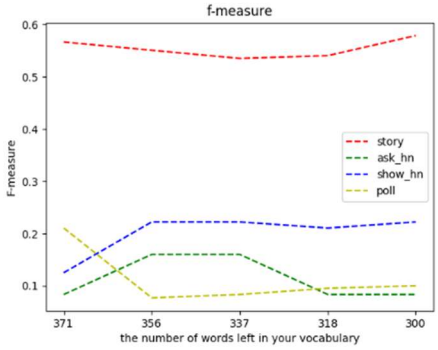


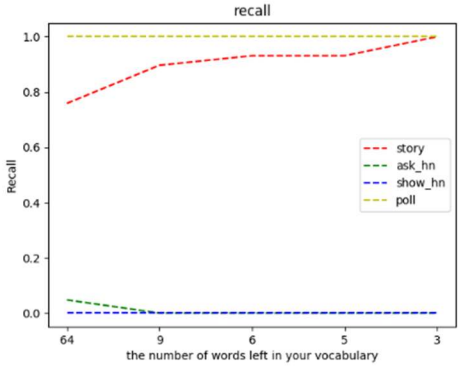


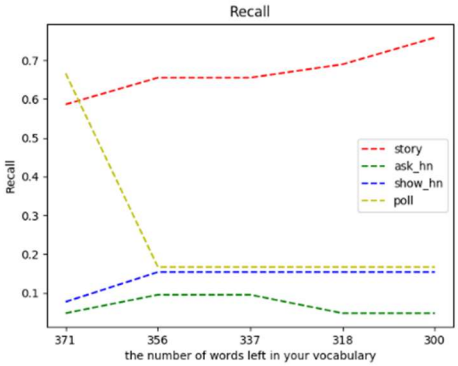


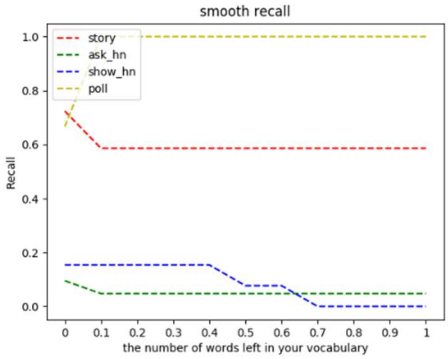


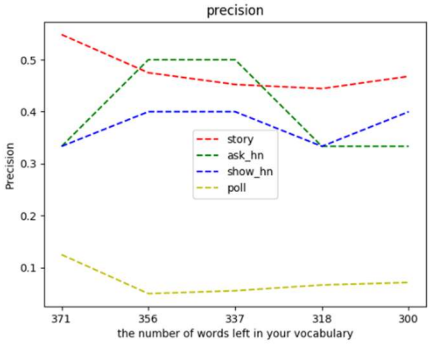


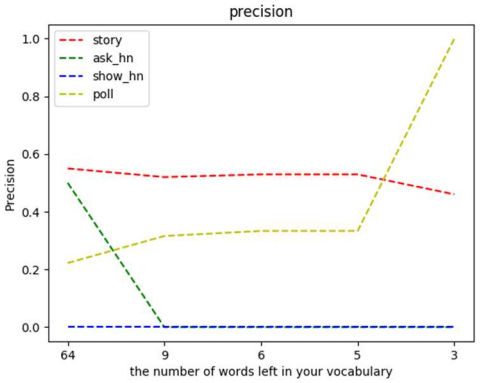


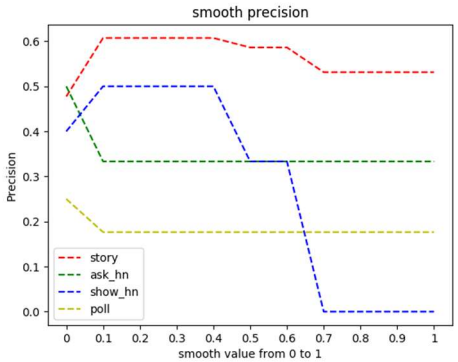












We can see that it is more correct to identify ‘poll’,’show_hn’,’ask_hn’. However, the accuracy is much lower than big dataset.

Secondly, if we have more time and energy, we would try to filter more efficient n-gram. For example, we can know Mr. Bob is a person and then collect it as a corpus.

Moreover, we only try to process English words in this experiment, and we remove word depending on English gramma. How about the dataset of other language(such as Chinese, French).

References

1. Multinomial Naïve Bayes Classifier for Text Analysis, https://towardsdatascience.com/multinomial-naive-bayes-classifier-for-text-analysis-python-8dd6825ece67
2. A practical explanation of a Naïve Bayes classifier, https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/
3. Wikipedia, Lemmatisation, https://en.wikipedia.org/wiki/Lemmatisation
4. Blog, 6 practices to enhance the performance of a text classification model, https://www.analyticsvidhya.com/blog/2015/10/6-practices-enhance-performance-text-classification-model/
5. Wikipedia, Additive smoothing, https://en.wikipedia.org/wiki/Additive_smoothing
6. Quora, What is Laplacian smoothing and why do we need it in a Naïve Bayes classifier, https://www.quora.com/What-is-Laplacian-smoothing-and-why-do-we-need-it-in-a-Naive-Bayes-classifier
7. IJCSMC Journal, Vol. 3. Issue. 10, October 2014, page 869-878, https://www.academia.edu/9040601/NAIVE_BAYES_CLASSIFIER_WITH_MODIFIED_SMOOTHING_TECHNIQUES_FOR_BETTER_SPAM_CLASSIFICATION
8. Blog, Spam filter classifier, precision and recall, https://zhuanlan.zhihu.com/p/32300580 (Chinese website)
9. Blog, Classification model performance evaluation – Accuracy, Precision, Recall, F-Score, https://zhuanlan.zhihu.com/p/37246394 (Chinese website)