Learning Multi-Label Topic Classification of News Articles

Zach CHASE
Nicolas GENAIN
Orren KARNIOL-TAMBOUR

***Abstract*—We consider the problem of learning to classify the topic of news articles for which there are multiple relevant topic labels. We analyze the shortcomings of a number of algorithmic approaches, including naive bayes, and develop two alternate approaches to solving the problem. In particular, we assess the performance of a binary classifier approach in which we learn a set of one-versus-all naive bayes binary classifiers, one for each label class. We also develop and analyze the performance of two novel algorithms derived from the popular tf-idf weighting scheme, and motivated by the goal of constructing a learning model that is more similar to the way human readers may classify article topics.**

I. Topic Classification of News Articles

Classifying the semantic content, or topic, of text is one of the critical problems in natural language processing, information retrieval, artificial intelligence and machine learning more broadly. Newspaper articles provide a particularly good opportunity for learning such classifications, as the semantic content of articles is generally coherent, and large, open source corpuses of labelled news articles exist and are easily accessible.

There is also a fair deal of interest in classifying news article topic for specific applications and research. News article topic classification can enable automatic tagging of articles for online news repositories and the aggregation of news sources by topic (e.g. google news), as well as provide the basis for news recommendation systems. More broadly, given the social and political impact of news and media, communications specialists and other social scientists - including our collegue Rebecca Weiss at the Stanford Communications department, who suggested this project to us - are particularly interested in analyzing news data programmtically to uncover patterns and biases in news production, content and dissemination.

A complicating factor is that news articles often fall under multiple topic labels. An article about a transfer in ownership of the Chealsea football club, for instance, might be labelled under Sports, Business, and World News. Humans can recognize and correctly provide multiple relevant labels for an article, but can a machine learning system get similar results?

II. Data

The New York Times has made a large corpus of its archive publicly available, with 5 years of articles hand tagged for various attributes of an article. Each article is provided along with a set of relevant taxnomic classifiers, which are essentially content labels. We chose to focus on learning only a small subset of the labels, 9 in total, corresponding to a number of major news topics: Business, Arts, Technology, Style, Books, Home and Garden, Sports, Science, and Health. Even when considering only 11 of many labels, most of the articles in our dataset had multiple tags, and some had many: the average number of articles per tag was 1.65, while the max was 6. Note that we also found that a fair number of the articles in the corpus were mislabelled - see analysis for more detail.

III. The Multi-Label Problem

A. Does it make sense to use Naive Bayes?

Consider the following corpus of articles, corresponding to the following priors:

label	article content
(1)	dog dog cat
(2)	animal object
(3,2,1)	dog vet

priors
p(1)=0.5
p(2)=0.5
p(3)=1/3

Obviously, we have that $p(1) + p(2) + p(3) > 1$ since our sets are not mutually exclusive - so computing class priors using Niave Bayes will not work. One solution to address this problem might be to use the exclusion-inclusion principle in order to make all our class sets mutually exclusive. By creating new classes, we can create a representation of our data as as a set of non-overlapping sets. This will allow us to have a set of class priors that both sum to 1 and represent the world faithfully. However, while this solution will

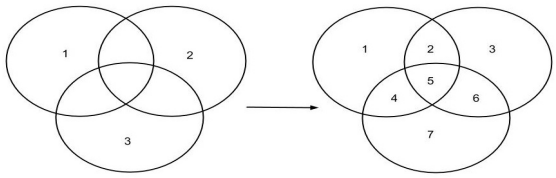

Figure 1. Using the exclusion-inclusion principle

allow us to model the world accurately, the introduction of additional classes and priors represents a significant increase in complexity. For our case of only 9 categories, we will already have $2^9 - 1$ classes to consider, i.e. just over 500 different class priors to compute - so this algorithm runs in exponential time. Additionally,

because the increase in label classes effectively splits the dataset across the set of new priors, we are providing far fewer training examples to each of the new mutually exclusive priors when compared to the 'old' ones - so this approach will also require far larger data sets to train on.

A common alternative approach to solving the mutual exclusivity problem is to learn independent binary classifiers for each class of labels in a dataset [1]. We now turn to an analysis of this approach.

IV. One versus all Naive Bayes classifiers

In the binary classifier approach, a binary classifier is learned for each class of labels, and overlaps between labels are essentially ignored. Each binary classifier constructs a prior on the relevant class it classifies, and posterior conditional probablities for each word are computed for two classes each time, corresponding to the probablity that a word was drawn from an article from class A, or from any article strictly outside the class. When presented with a new test article, the system runs each classifier and decides class by class if an article belongs to it, or not. The following chart provides an illustration of the approach used for three class labels: Since learning each binary classifier is an independent

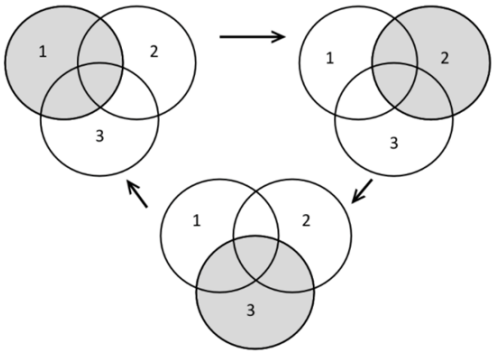

Figure 2. One versus rest Naive Bayes classifiers principle

task, the binary classifier approach is computationally scalable. It also produces fairly accurate results. The table below summarizes the classification error of the binary classifiers on three feature sets: full article text, lead paragraph, and article headline. The model's average labelling error across classes for each feature, respectively, was 3.1%, 4.25%, and 6.2%. Interestingly, the binary classifiers are fairly accurate at predicting the correct labels even when presented only with the headline of an article.

While the binary classiffiers yielded fairly robust results, we found two primary limitations of the approach that led us to explore an alternative model.

	Full Text	Lead Paragraph	Headline
Business	3.8%	5.8%	8.0%
Arts	4.6%	5.0%	12.4%
Technology	5.5%	7.0%	5.9%
Style	3.5%	4.0%	6.1%
Books	2.4%	4.2%	4.4%
Home	0.9%	1.8%	1.6%
Sports	1.2%	2.0%	8.9%
Science	2.7%	3.2%	3.2%
Health	3.7%	5.3%	5.9%

Figure 3. Performance of binary classifiers on three feature sets: full article text, lead paragraph, and article headline.

First, the binary classifiers cannot be directly modified or improved, and an improvement of the model would likely have required us to instead implement an SVM. Second, and more critically, we wondered if it might be possible to construct a learning model that more closely approximates the way human readers may perform a similar classification task.

V. Solving the problem the way human readers do

How do human readers quickly and accurately classify the content of an article? Our own introspection suggests that one way readers do this is by scanning an article and picking up on 'tip-off' words – words that are highly indicative of the article belonging to a particular topic. As a simple example, if a quick scan of an article returns the word 'Obama', a human reader might think the article relates to politics, but have somewhat low confidence in that classification; if the scan also returns the words 'policy' and 'congress', the reader is highly likely to classify the article as relating to politics, and moreover, to be fairly confident of that classification. This idea and the notion of 'tip-off' words suggested to us that fairly robust multi-label classification should be achievable with only a limited set of high-information words, and moreover, without access to any explicit priors on class labels. Consequently, we attempted to develop a model that could find such a set of high-information words and use it to classify multiple topic labels.

VI. A derivative of the Term Frequency - Inverse Document Frequency model

A. A standard implementation of the model

Since we are looking for high-information words, we turned to information retreival and search for tools. Topic classification can easily be thought of as a search/information-retreival problem: given a query

(in our case, an article), which result (in our case, topic label) is most relevant to the query? A common weighting scheme used in search and information retrieval is term frequency–inverse document frequency (tf-idf), a numerical statistic that reflects how important a word is to a document in a collection or corpus. It is particularly used in web-search to rank pages by keyword. We construct a simple variant of this scheme for our problem.

The first element is the term frequency (tf), i.e. the number of times that token w occurs in an article a, summed across all the articles in a particular class:

$$\forall w,\ tf(w) = \sum_{articles\ a} \frac{\sum_{j=1}^{n} 1\{w = x_j^{(a)}\}}{\sum_{j=1}^{n} x_j^{(a)}}$$

Intuitively, because we are summing over all the articles in a given class, the tf-score captures both the frequency of a word in each article and the frequency of a word across articles in a given class; words that appear with high frequency across many articles in a class will be assigned particularly high tf-scores. Next, the inverse document frequency (idf) is a measure of whether a particular term is common or rare across the whole corpus of articles. It is obtained by dividing the total number of words in the corpus by the count of the instances of the particular word in the data, and then taking the logarithm of that quotient:

$$\forall w,\ idf(w) = log\left(\frac{\sum_{articles\ a,\ words\ j} x_j^{(a)}}{\sum_{articles\ a} x_w^{(a)}}\right)$$

Intuitively, words that occur frequently in a specific class but infrequently in the corpus in general constitue high-information words for a given class, while those that appear often across the corpus are less informative. Multiplying the tf-score of a word by its itf-score allows us to take both aspect into consideration. Thus, we the compute for each token the tf-idif score: $\forall w,\ tfidf(w) = tf(w) * idf(w)$.
For each class we short-listed the 100 most relevant words - the highest information words for each class - and used only them to score our testing examples. For a given testing example, we compute a score per category: each time a token in the testing article also appears in the short-list of the category, we add the score of the word to the total score for that category:

$$score(category\ c) = \sum_{word\ in\ w} tfidf(w)$$

Figure 4 shows a plot of the top 100 TFIDF values for Business. The shape of the function reflects the intuition that one can capture very high information about a class with only a few words - and in fact the plot of the next 500 words shows a further exponential drop in tdidf values.

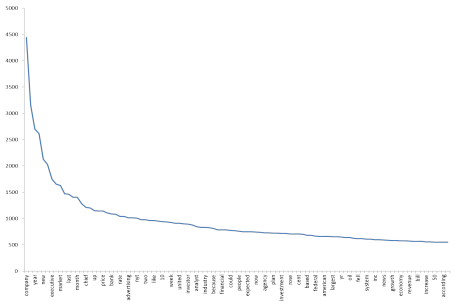

Figure 4. Top 100 TFIDF values for the Business category

B. Learning a threshold

We now have a final score per category for our test article, but we still have to make a decision on whether it belongs to each class or not. For this we can use two different machine learning techniques: kmeans, which we implemented, and logistic regression, which we did not but describe briefly below.

1) Threshold with k-means: The result of the scoring of a testing article often looks like this:

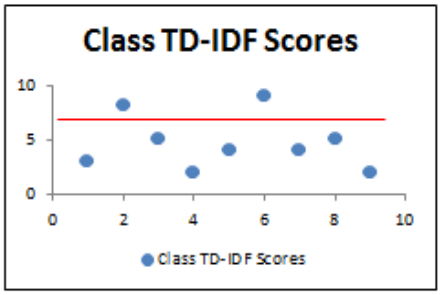

Figure 5. Idealized example of learning a selection threshold with k-means

Our goal is to cluster these points in two different categories: ”high” scores and ”low” scores. For the high scores, we will predict 1, meaning that the article belongs to those classes, and for the low scores we will predict 0, meaning that the article does not. An advantage of the k-means approach is that since threshold selection for each article is independent, we are able to select a threshold and make a reasonable prediction from the very first article we score. This feature of k-means is also satisfying from the perspective of finding a learning method that is more similar to the way human readers classify - we can predict scores immidiately and independetly of previous

thresholds for other articles.

A problem with this method is that the ranges of the scores across topic categories are often not on the same scale. Categories with a large number of articles tend to have higher tfidf values than categories with a smaller number of articles, since we are summing the tf-scores for each word over every article in each class. Thus, the tfidf score for classes in which the number of articles is a few times larger are often significantly higher. This jeopardizes the underlying assumption of k-means: that the scores are comparable. This also explains the relative magnitdue of false positives our system generated for larger categories such as Business, Sports, and Arts - see results section for more detail.

2) Threshold with logistic regression: Here we take a very different approach. Instead of learning a selection threshold over tfidf scores for each class, article by article, we instead learn over a great number of training examples a threshold per category over which we decide a label (i.e. a prediction of y=1) needs to be applied. This threshold can be learned using logistic regression. One of the problems with this method is that the tfidf scores are dependent on the size of the article: the longer the article, the higher the score. Consequently, using logistic regression to learn a threshold would once again require some kind of normalization of our computed scores. Lastly, a downside of logistic regression is that it potentially requires exposure to a very large number of articles before we can start making meaningful threshold predictions - such that we lose our ability to mimic the capacity of human readers in classifying new articles indepedently of old ones.

C. Results

Our tfidf model had overall worse performance on calssifying topic labels than did the binary classifier approach. The table below shows the classification errors produced for each class, broken out into false positives - cases in which we mistakenly predict a label - and false negatives, or cases in which we neglect to predict a label for a given article.

Interestingly, virtually all of our errors turned out to be false positives, and were highly concentrated in three classes: Business, Arts, and Sports. Our analysis of the tfidf values in the false positive cases showed that the relative size of the classes was resulting in very large tf values in comparison with other classes, and thus degrading the efficacy of k-means in learning a selection threshold.

We explored two methods of fixing this problem, based on two sources of the error. Our first approach was to attempt to normalize the scores by dividing the tf-score for each class by the number of articles in that class. This actually resulted in worse performance, so we tried a different normalization scheme and instead divided the tfidf score for each category by the max score observed for that category, such that we could map tfidf scores for each category to a normalized scale ranging from 0 to 1. This too produced worse performance. Through this trial and error process, we realized that normalizing the class sizes was actually a mistake - the difference in class sizes captured through the summation over all the articles in each class in some ways also captured something akin to a prior on each class, and k-means needs a meaningful difference between outputs to learn a good selection threshold.

Error Type	False Positiv	False Negati	Total Error
Technology	2.8%	3.4%	6.2%
Sports	24.0%	0.0%	24.0%
Home	2.8%	1.2%	4.0%
Business	26.6%	0.2%	26.8%
Books	2.6%	4.8%	7.4%
Health	4.2%	3.0%	7.2%
Science	2.8%	3.0%	5.8%
Arts	27.6%	0.2%	27.8%
Style	5.0%	3.2%	8.2%

Figure 6. Error determined using top 100 td-idf words for each class, and tested on 500 articles

Our second method focused on trying to eliminate the impact of some of the noisier terms in our top tfidf scores for the biggest classes in particular. Words that were very common across our corpus and had very low idf-scores were still showing up in the top 100 tfidf words for large classes - see figure 6 - because they were still appearing in a very large number of articles for the biggest classes. So in particular, we tried to come up with a scheme which would increase the relative weighting of the idf-score component relative to the tf-componenet of the tfidf-score, which led us to a variation on our origional model.

D. A variation on tf-idf

A closer look at the top 100 tfidf-scores for the three largest classes indicated that at least a large part of the false positive classification error was caused by the persistence of high frequency but low information words in our top words list. Figure 7 shows the top 20 words for the Business class by tfidf-score. Note that while most of the words are extremely high information and very relevant to the class - words such as 'company', 'market', and 'share' - a number of words, marked in read, are frequent but not particularly indicative of business articles. The extremely high tfidf scores of

words such as 'year' and 'month' turned out to be largely responsible for the high rate of false positives. Indeed, if one of these words appear in the article, the article is very likely to be tagged business whereas the presence of this word actually doesn't mean much.

Because we wanted to avoid removing these words artificially from our top 100 tf-idf lists and were concerned that other low-value words might take their place if we did, we devised a scoring scheme that would place a heavier emphasis on the idf-score of a word such that words with low idf scores would be unlikely to have high tf-idf scores. In the original idf model, many of the high tfidf but low information had extremely low idf-scores, reflecting their generally high frequency across the corpus.So we needed to modify the tfidf function such that words with low idf values would be mapped to very low overall tfidf scores. On the other hand, we wanted an idf function that ensured that words with high idf values stayed in $O(log(\ inverse\ document\ frequency))$. We handcrafted the following function with the desired properties: The resulting model is expressed by the formal equation:

word	tfidf value
company	4439,83416
percent	3160,94197
year	2702,08176
million	2614,25251
new	2128,63562
business	2028,49484
executive	1749,33569
billion	1656,34608
market	1628,28785
share	1470,06762
last	1459,24475
stock	1407,06511
month	1403,32911
group	1280,50026
chief	1214,14536
sale	1202,07456
up	1151,39047
amp	1144,31421
price	1144,18648
york	1106,19767
bank	1086,6829
time	1079,53584
rate	1039,92085
service	1039,00863

Figure 7. Top 20 TFIDF words and values for the Business category

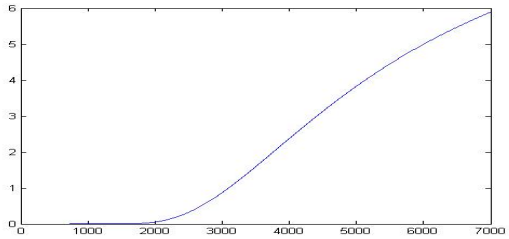

Figure 8. Error determined using top 100 td-idf words for each class, and tested on 500 articles

$$\forall w,\ idf(w) = exp\left(\frac{-\alpha}{\left(\frac{\sum_{articles\ a\ words\ j} x_j^{(a)}}{\sum_{articles\ a} x_w^{(a)}}\right)^2}\right) * log\left(\frac{\sum_{articles\ a\ words\ j} x_j^{(a)}}{\sum_{articles\ a} x_w^{(a)}}\right) \quad (1)$$

Near 0, the term inside the exponential determines the shape of the funcion, while near $+\infty$, the term inside the log is prevalent. This approach yielded the following results:

Error Type	False Positive	False Negative	Total Error
Technology	11.8%	1.4%	13.2%
Sports	11.4%	0.2%	11.6%
Home	11.2%	0.6%	11.8%
Business	14.2%	1.0%	15.2%
Books	11.6%	1.4%	13.0%
Health	11.4%	1.8%	13.2%
Science	10.8%	1.6%	12.4%
Arts	12.2%	2.6%	14.8%
Style	14.2%	0.6%	14.8%

Figure 9. Error determined using top 100 td-idf words for each class, and tested on 500 articles

The results are better on average, but the errors are now evenly distributed across the classes. Nevertheless the results are still far from the performance of Naive Bayes, which leaves us the opportunity for further investigation.

On the whole our research validated the common approach of using binary-classifiers to learn multi-label topic classifications for new articles. The tfidf approach captures some interesting aspects of the intuition behind how people may classify news articles, but we were not able to lower the error produced by the tfidf model sufficiently to make it practically competitive with the binary classification scheme. Future research might look into alternate methods for scoring functions based on tfidf and the notion of finding high information words to classify multi-label articles.

REFERENCES

[1] Rong-En Fan and Chih-Jen Lin. John W. Dower *A Study on Threshold Selection for Multi-label Classification.* Technical Report, National taiwan University, 2007.

[2] Arzucan Ozgur, Levent Ozgur, and Tunga Gungor. *Text Categorization with Class-Based and Corpus-Based Keyword Selection.* Proceedings of the 20th international symposium on computer and information sciences, Springer-Verlag (2005), pp. 607-616. E. H. Norman