# *How hot will it get? Modeling scientific discourse about literature*

Natalie Telis, CS229
ntelis@stanford.edu

## Project Aims

Many metrics exist to provide heuristics for quality of scientific literature, some of which are determined before publication, such as impact factor of journal, or affiliations of authors. Others, like citation count, are not. None of these metrics take into account the amount of scientific discourse generated by an article. I aim to create a classifier that does. In particular, in the wake of the rise of social media, more and more scientific discourse has taken place in public, through blogging, facebook posts, and even tweets about papers. Can we predict how much discourse a paper will generate?

## Data preparation

I have annotated each of the 445,615 papers indexed within NCBI's PubMed database in the last 6 months with a tweeted-score for each paper $i$ defined as

$$t_i = \alpha \times \text{count}(\text{tweets}_i) + \beta \times \text{count}(\text{retweets}_i)$$

A **tweet** is defined as a message, with or without content, posted on twitter which can be linked (by URL, which can be used to identify DOI or PubMed database ID) to a paper. In assigning my scores, I use $\alpha = 1$. A **retweet** is a tweet indicating that it has been sourced from a previous tweet, with **no additional content added**. E.g., a retweet which adds commentary – such as "rt @scientist_twitter Nice paper! `http://link.db`" – is considered a tweet of its own, whereas a retweet with no additional commentary, "rt @scientist_twitter `http://link.db`", is considered a simple retweet. I set $\beta = 0.5$.

After calculating the tweeted-score $t_i$ for our library, although the data is not normal, I calculate a $z$-score $z_i = \frac{t_i - \mu_t}{\sigma_t}$ for every tweet score. I do this as otherwise the score distribution is extremely noisy, varying from 0 across most of the set to scores in the hundreds of thousands for a scant few papers.

Once I have the normalized tweeted-scores, I first aimed to understand the distribution of discourse. Unfortunately, across the body of papers, just shy of $75,000$, or about 17% of papers actually receive even a single tweet. This is understandable, as many papers come from lower impact journals (average IF in the set is 3.12; however, there is frequently lossy data about exact impact factor due to new journals, which are therefore assigned an impact factor median over all journals, which is 1.0 – this may bias estimations of average impact factor in a subset of papers).

Moreover, some tweets may come from spammers, or from laypeople. However, manual inspection suggests that the fraction is low; moreover, most tweets in that category appear to be retweets and are downweighted by $\beta < \alpha$, which suggests that they likely do not inflate the scores enough to affect the quality of the data.

# Methods

## Methods used

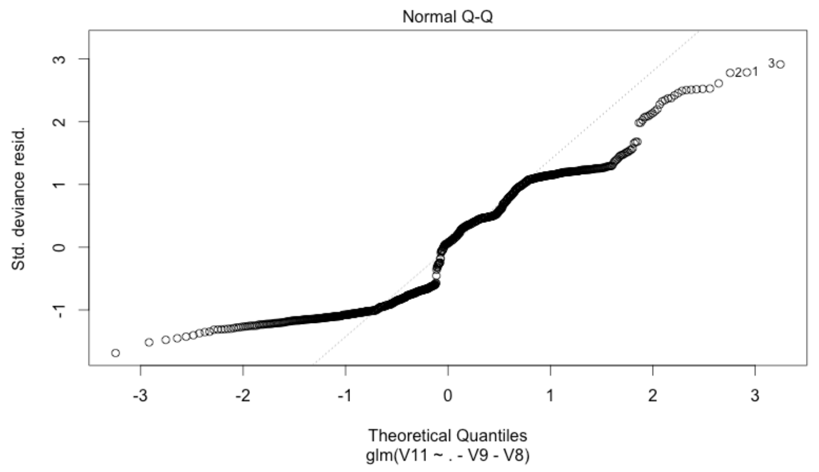
**Topic assignment:** Latent Dirichlet Allocation[1] (LDA) is used for topic assignment. Our inferencer was trained on 1.5 million papers from NCBI's PubMed with a dictionary consisting of all words in the titles and abstracts of those papers, with stop words removed. The training process instantiates $k$ latent topics, each with a Dirichlet distribution over the dictionary, and is refined through iterative Gibbs sampling.
**Naive Bayes:** Naive Bayes is a binary classification scheme which can use binary data to predict a binary response variable. Probabilities of each word in a dictionary occurring in a training example in one category (without loss of generalization, labeled "1") or in the other (labeled "0") are defined according to Bayes' rule, with 1 added in the numerator, and the size of the vocabulary on the denominator, to prevent undefined values in the case that a word was not observed in the training dataset (Laplace smoothing):

$$p(w_i|\text{state}) = \frac{\text{counts}(w_i, \text{state}) + 1}{\text{total} + |\mathbb{V}|}$$

Assuming each word occurs independently of another, the probability of a document occuring given a response variable state is $\prod_{i=1} p(w_i|\text{state})$. This probability can be easily calculated for both possible states, and the higher probability corresponds to the predicted state. I implemented this method myself.
**Support Vector Machines (SVM)**: The support vector machine relies on data points close to the margin between two classification groups to identify maximal margin hyperplanes that partition two classifications. I used the `R` package `LiblineaR`'s implementation.

## Preliminary approach

I began by fitting a generalized linear model to the standardized twitter z-scores. However, upon plotting, I quickly saw that this model would suffer significantly because of the number of papers with tweets.



This QQ plot suggests that although the fit of the linear model is pretty okay for papers with a median amount of tweets, the very long tail of papers with no tweets and the variable high tail suggests that a linear model is not ideal for approximating those papers.

Because of concerns about the reasonability of fitting a linear model to a dataset in which a majority of the data is equal to zero, I decided to put off plans for a linear model and instead to consider binary classification – "high twitter activity" (1) versus "low twitter activity" (0). I annotated each paper in my set according to this paradigm. I considered papers with $\geq 2$ standard deviations tweet scores to be

I generate a training set and a testing set from my data. Although this does not reflect the actual proportions in the dataset, for demonstrative purposes, I selected 50% papers which do have tweets (categorized as 1s), biasing towards those which get a lot of tweets, and 50% papers which do not have any tweets at all (categorized as 0s). I do this by random assignment, so my training set and testing set are usually slightly uneven (e.g., 850 and 1000) - however, by Fisher's exact test I determine that the assignment of papers with high tweets was not distinct between the training and testing sets.
I then define a dictionary based on words which appear more than once in the training and testing sets, and use this dictionary to generate word counts for both training and testing sets. Then, I attempt to train several models to create a classifier. Both Naive Bayes and SVM showed extreme signs of overfitting, so I readjusted my approach.

## Classification without topic partitions

I first attempted to use a Naive Bayes approach for classification. The Naive Bayes approach is good for this kind of problem, as it takes into account the class prior during assignment, and also aggregates potentially weak effects into a stronger signal of class membership. However, due to the extreme sparseness of the word matrix for high- and low-activity tweeted papers, this approach was poor (see below) – no better than random. I decided to try another approach, focusing more on the borderline cases – a support vector machine.

The SVM I trained suggested signs of serious overfitting. I theorized that this overfitting was a result of poor concordance between training high-activity papers and testing high-activity papers – as there is a deal of underlying structure in the documents, such as differing scientific subfields, and there are very few highly-tweeted papers, the sets of high activity papers might differ from sample to sample.

Therefore, I repeated my previous analyses and data-gathering methods, but I first restricted my data set by using Latent Dirichlet Allocation[1] topic assignments to create a soft classification for each paper. Then, restricting to papers only within a particular topic (there are a total of 150 topics), I repeated my training and testing procedure from above.

## Results

Several datasets for training were used during the course of the project, denoted below:

- Schema A: A random training and testing set of 2000 papers subsampled from the corpus of 445,615 total papers annotated in the last 6 months.
- Schema B: A training and testing set made by randomly assigning all papers with highest probability membership in a chosen topic (material science and technology) to either training or testing. ($n = 3750$)
- Schema C: Like schema B, but limiting membership in training and testing sets to papers with probability $> 0.6$ of belonging to material science and technology (more "conclusive" assignment). ($n = 1875$)

| Technique | Data used | Features included | Training error | Testing error |
|---|---|---|---|---|
| Naive Bayes | A | All | 0.5 | 0.5 |
| Naive Bayes | B | All | 0.08 | 0.20 |
| Naive Bayes | C | All | 0.08 | 0.17 |
| Naive Bayes | C | Words in $\geq 1$ training document | 0.07 | 0.17 |
| SVM | A | All | 0.001 | 0.49 |
| SVM | B | All | 0.001 | 0.27 |
| SVM | C | All | 0.02 | 0.28 |
| SVM | C | Words in $\geq 1$ training document | 0.0 | 0.22 |

## Discussion

I believed much of the challenges in training resulted from the breadth of possible subject material. This was confirmed by the vast decrease in overfitting once I restricted my training to a particular topic. In the beginning, the results from both Naive Bayes and the SVM models suggest serious overfitting. This is likely because the relevance of training to testing data was initially very thrown off by subsampling (as a sample of 1000, done twice, from 75,000, is not guaranteed to be representative at all).

These results suggest that my model could continue to improve with intelligent matching of training and test data. The progressive tightening of the gap between training and test error with the 3 different training and testing data generation schema suggests that by more and more intelligent training strategies I could hope to improve the model even beyond this.

One way in which I plan to implement this is to consider the full topic probability vector rather than simply the most probable assignment. This would help intelligently choose training data that does not confound, and there is a lot of correlation structure in similar joint assignment (e.g., many papers assigned with high probability to one topic are often assigned with moderate probability to only one or two others; partitioning those sets in those ways may further pare down error).

I also plan to consider improvements on my learning techniques. For example, a multinomial Naive Bayes model may be more effective than one based simply on binary word occurrence, in the "bag of words" model. Likewise, I could potentially improve the model by including binary metadata, such as "high" or "low" impact factor.

One question of interpretation of these results is methodological: what association do words seem to evoke, and why do certain words mean papers get more tweets? In particular, for Material Science and Technology, the topic showcased in this report, the two most leading terms are "nanotube" and "pacemaker". This suggests to me that there are particular words that are indicative of particular areas in a field that are hot. This suggests that similar prediction, with a much longer spread of social media data, could even lead to longitudinal trend identification in terms of interests and directions in a particular field which are seen as interesting or hot.

Another final methodological question is to note that tweets may indicate outrage, curiosity, confusion, or amusement – not just interest. Since our repository also contains tweet text as well as tweet authors, perhaps sentiment modeling (with some correction for longitudinal average sentiment) could help partition papers into "high positive", "high negative", and other sentiment and activity co-categories.

In conclusion, although the current model performs well, there are many open problems in the field of prediction of discussion around a text to continue to explore, and I hope to be able to by expanding on the techniques outlined in this project.

## References

1. *Latent Dirichlet Allocation.* Blei, et al. Journal of Machine Learning Research. 3 (2003) 993-1022.