Tackling the Poor Assumptions of Naive Bayes Text Classifiers

A Naive Bayes Improvement

Parminder Singh [2016MT10630]

Harsh Kumar [2016MT10629]

Anshuman Shrivastava [2016MT10620]

Problem -1

Naive Bayes selects poor weights

When one class has more training examples than an-other, Naive Bayes selects poor weights for the decision boundary. This is due to an under-studied bias effect that shrinks weights for classes with few training example.

To balance the amount of training examples used per estimate, we introduce a “complement class”formulation of Naive Bayes.

Problem -2

Features Independence

As a result, even when words are dependent, each word con-tributes evidence individually. Thus the magnitude of the weights for classes with strong word dependencies is larger than for classes with weak word dependencies.To keep classes with more dependencies from dominating, we normalize the classification weights.

Problem -3

Multinomial Naive Bayes

In addition to systemic problems, multinomial Naive Bayes does not model text well. We present a simple transform that enables Naive Bayes to instead emulate a power law distribution that matches real term frequency distributions more closely.

Multinomial Naive Bayes

Assumptions

Multinomial Naive Bayes models the distribution of words in a document as a multinomial. A document is treated as a sequence of words and it is assumed that each word position is generated independently of every other.

Modeling and Classification

For classification, we assume that there are a fixed number of classes, $c \in \{1,2,...,m\}$, each with a fixed set of multinomial parameters. The parameter vector for a class c is $\theta_c = \{\theta_{c1}, \theta_{c2}, ..., \theta_{cn}\}$, where n is the size of the vocabulary, $\sum_i \theta_{ci} = 1$ and θ_{ci} is the probability that word i occurs in that class. The likelihood of a document is a product of the parameters of the words that appear in the document,

$$p(d|\vec{\theta}_c) = \frac{(\sum_i f_i)!}{\prod_i f_i!} \prod_i (\theta_{ci})^{f_i},$$

Classification

By assigning a prior distribution over the set of classes, $p(\theta_c)$, we can arrive at the minimum-error classification rule (Duda & Hart, 1973) which selects the class with the largest posterior probability,

$$l(d) = \operatorname{argmax}_c \left[\log p(\vec{\theta}_c) + \sum_i f_i \log \theta_{ci} \right] = \operatorname{argmax}_c \left[b_c + \sum_i f_i w_{ci} \right],$$

Parameter Estimation

Parameters must be estimated from the training data. We do this by selecting a Dirichlet prior and taking the expectation of the parameter with respect to the posterior. For details, we refer the reader to Section 2 of Heckerman(1995). This gives us a simple form for the estimate of the multinomial parameter,

$$\hat{\theta}_{ci} = \frac{N_{ci} + \alpha_i}{N_c + \alpha},$$

Substituting the true parameters in Equation 2 with our estimates, we get the MNB classifier,

$$l_{\mathrm{MNB}}(d) = \operatorname{argmax}_c \left[\log \hat{p}(\theta_c) + \sum_i f_i \log \frac{N_{ci} + \alpha_i}{N_c + \alpha} \right]$$

The prior class probabilities, $p(\theta_c)$, could be estimated like the word estimates

Correcting Systematic Errors

Skewed Data Bias

skewed data—more training examples for one class than another—can cause the decision boundary weights to be biased.

Each class has a binomial distribution with probability of heads ***θ= 0.25*** and ***θ= 0.2***, respectively.
We are given one training sample for Class 1 and two train-ing samples for Class 2, and want to label a heads ***(H)*** occurrence.

Class 1 $\theta = 0.25$	Class 2 $\theta = 0.2$	$p(\text{data})$	$\hat{\theta}_1$	$\hat{\theta}_2$	Label for H
T	TT	0.48	0	0	*none*
T	{HT,TH}	0.24	0	$\frac{1}{2}$	Class 2
T	HH	0.03	0	1	Class 2
H	TT	0.16	1	0	Class 1
H	{HT,TH}	0.08	1	$\frac{1}{2}$	Class 1
H	HH	0.01	1	1	*none*

$$p(\hat{\theta}_1 > \hat{\theta}_2) = 0.24$$
$$p(\hat{\theta}_2 > \hat{\theta}_1) = 0.27$$

"skewed data bias," directly results from imbalanced training data. If we were to use the same number of training examples for each class, we would get the expected result—a heads example would be more often labeled by the class with the higher rate of heads.

complement Naive Bayes

In estimating weights for regular MNB we use training data from a single class ***c***. In contrast, CNB estimates parameters using data from all classes except ***c***. CNB's estimate is

$$\hat{\theta}_{\tilde{c}i} = \frac{N_{\tilde{c}i} + \alpha_i}{N_{\tilde{c}} + \alpha},$$

Effective because - each uses a more even amount of training dataper class, which will lessen the bias in the weight estimates.

NOTATION - $N_{\tilde{c}i}$ -----> the number of times word ***i*** occurred in documents in classes other than c.
$N_{\tilde{c}}$ -----> the total number of word occurrences in classes other than ***c***
α_i & α -----> smoothing parameters,

The classification rule is

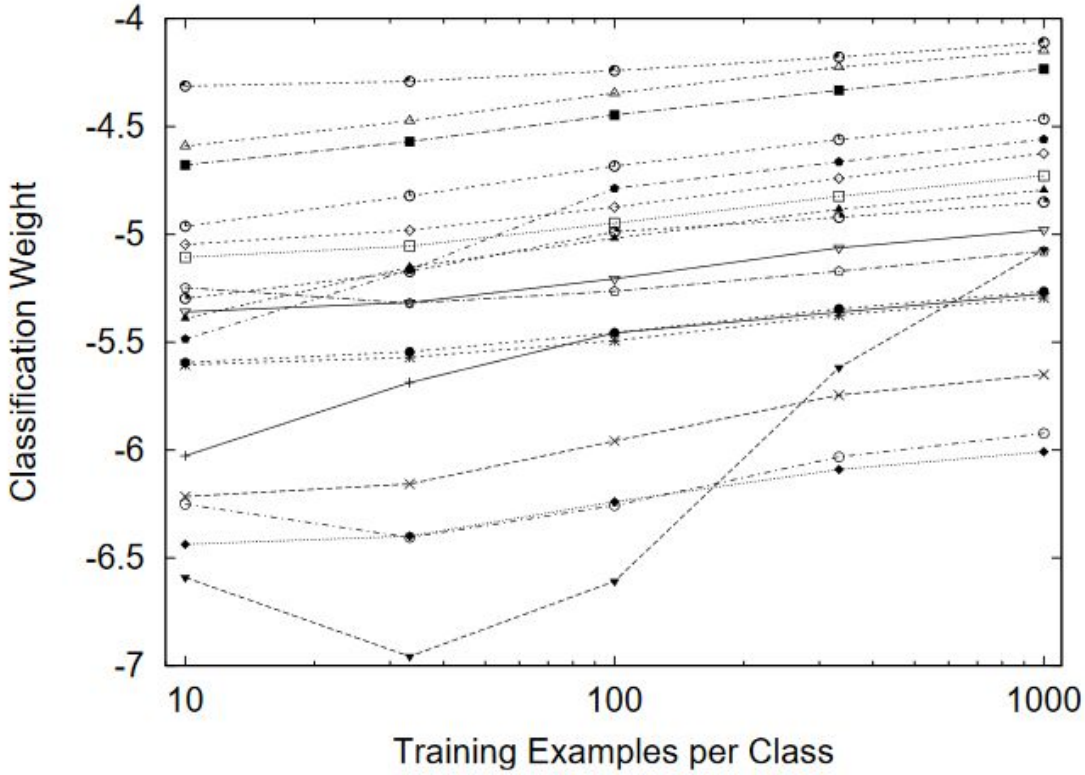
$$l_{\mathrm{CNB}}(d) = \mathrm{argmax}_c \left[\log p(\vec{\theta}_c) - \sum_i f_i \log \frac{N_{\tilde{c}i} + \alpha_i}{N_{\tilde{c}} + \alpha} \right]$$

The negative sign represents the fact that we want to assign to class c documents that poorly match the complement parameter estimates

Analysis - (A)

The regular NB weight estimates

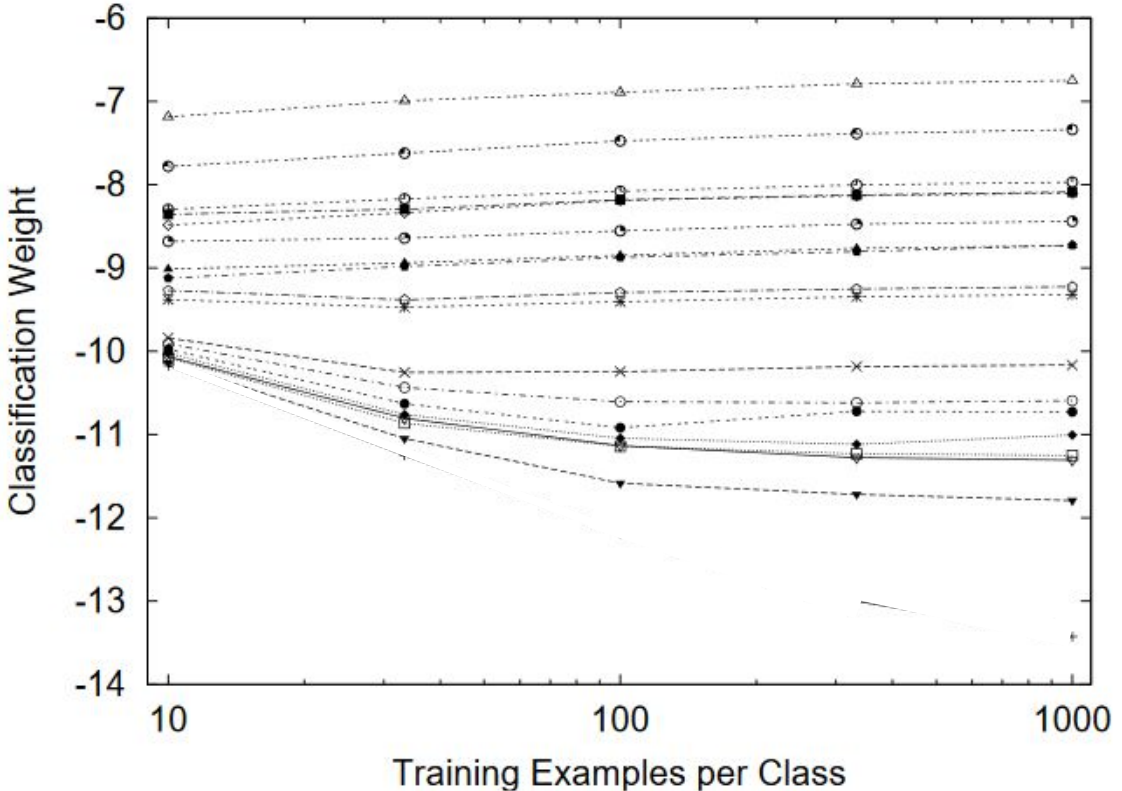
In particular, the word that has the smallest weight for 10 through 100 examples moves up.



Analysis - (B)

The complement weight estimates

The complement weights show the effects of smoothing, but do not show such a severe upward bias and retain their relative ordering.



Weight Magnitude Errors

Naive Bayes gives more influence to classes that most violate the independence assumption.
Differences in weight magnitudes are not a attempt to create greater influence for one class.

EXAMPLE -

San	Francisco

Boston

Since "San Francisco" and "Boston" occurs equally in their respective classes, a single occurrence of "San Francisco" will contribute twice the weight asan occurrence of "Boston."

CORRECTION-

Instead of assigning $w_{ci} = \log\theta_{ci}$ we assign

$$\hat{w}_{ci} = \frac{\log \hat{\theta}_{ci}}{\sum_k |\log \hat{\theta}_{ck}|}.$$

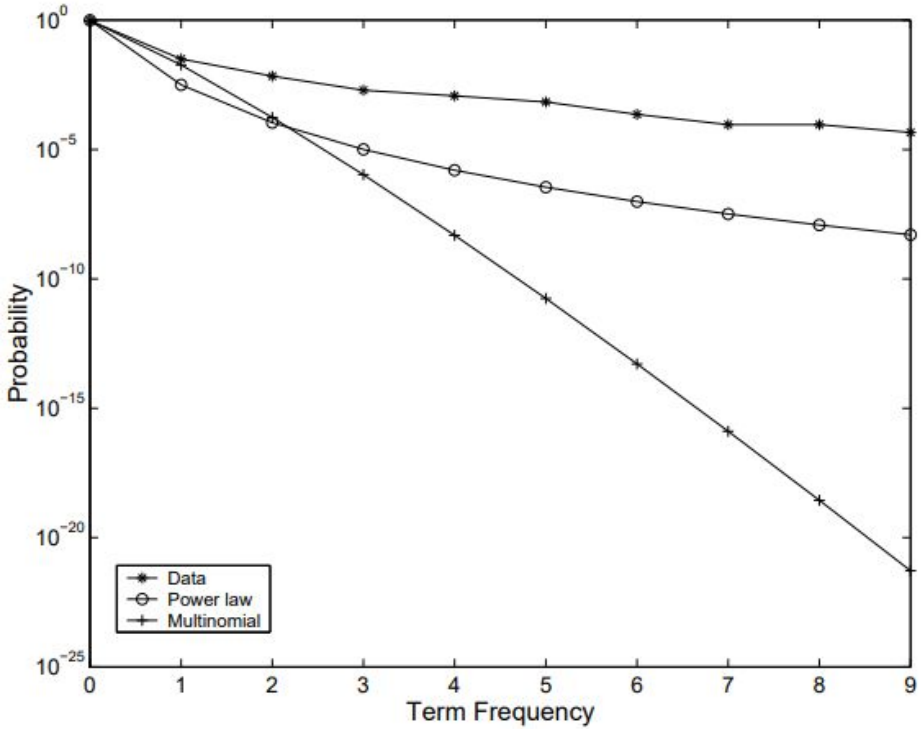
Modeling Text Better

Transforming Term Frequency

Term frequency distributions have a much heavier tail than the multinomial model expects.

REASON-

By transforming the data to be better suited for use with a multinomial model, we find significant improvement in performance over using MNB without the transforms

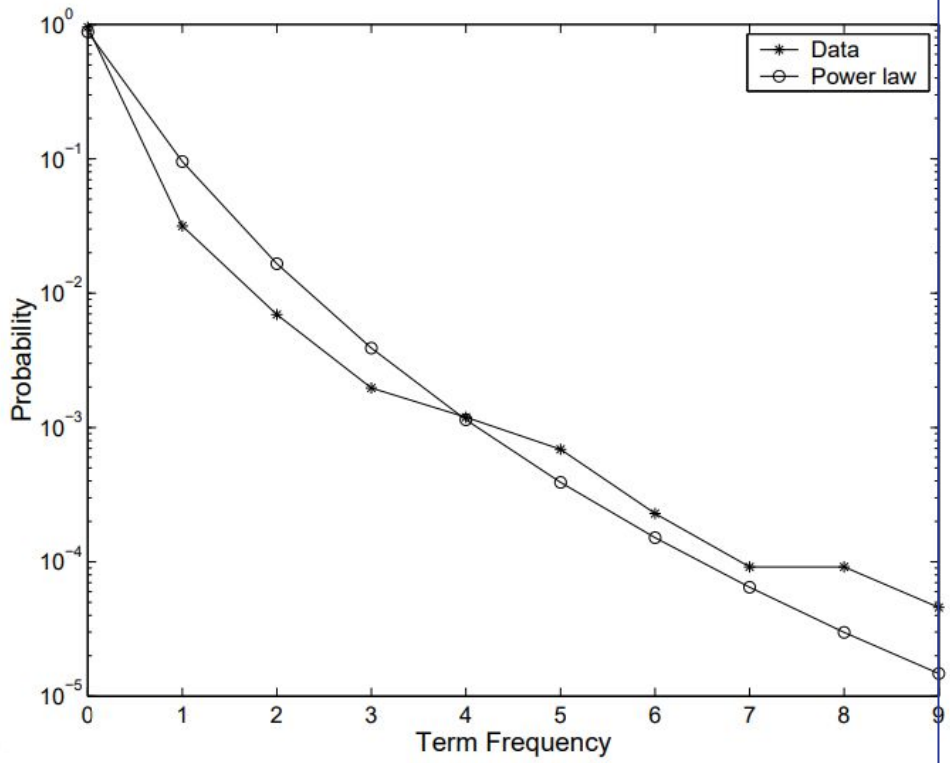


Transforming Term Frequency

Thus power law distribution is used instead of direct multinomial distribution , ***p(fi) ∝ (d + fi) log θ*** ,

where ***d*** has been chosen to closely match the text distribution.

The probability ***∝ θ^(log(d+fi))***. So, we can use the multinomial model to generate probabilities proportional to a class of power law distributions via a simple transform, ***fi' = log(d + fi).***



Transforming Document Frequency

Common words are unlikely to be related to the class of a document, but random variations can create apparent fictitious correlations. This adds noise to the parameter estimates and hence the classification weights and can sway over a decision. For this reason, it is advantageous to downweight these words. A heuristic transform "***inverse document frequency***", is used:

$$f_i' = f_i \log \frac{\sum_j 1}{\sum_j \delta_{ij}},$$

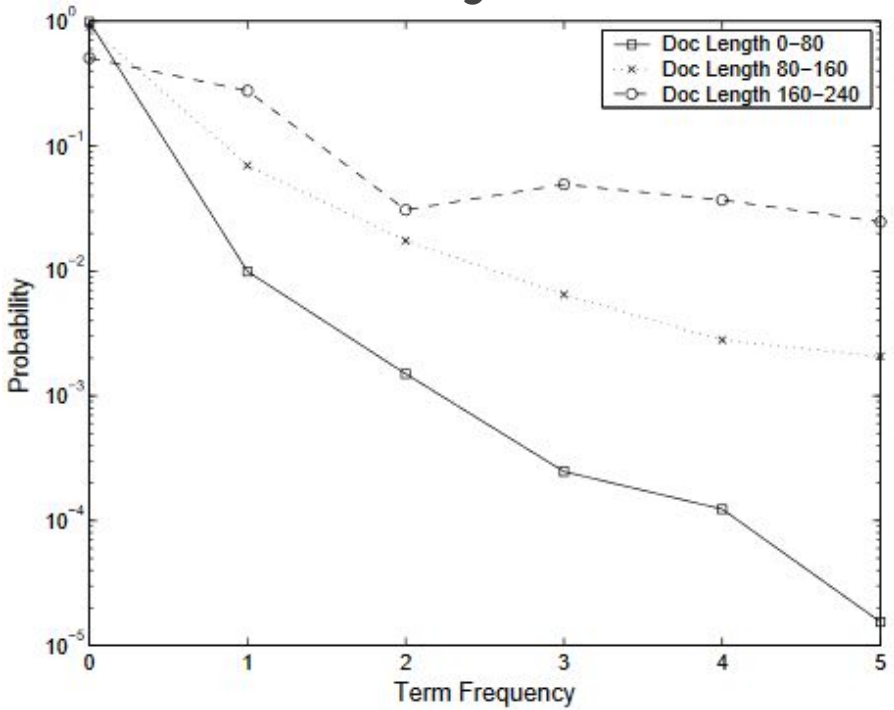
where δij is 1 if word i occurs in document j, 0 otherwise.

Transforming Based on Length

Empirical term frequency distributions for documents of different lengths

We discount the influence of long documents by transforming the term frequencies according to

$$f_i' = \frac{f_i}{\sqrt{\sum_k (f_k)^2}}.$$



FINAL ALGORITHM

- Let $\vec{d} = (d_1, \ldots, d_n)$ be a set of documents; d_{ij} is the count of word i in document j.
- Let $\vec{y} = (y_1, \ldots, y_n)$ be the labels.
- TWCNB($\vec{d}, \vec{y}$)
  1. $d_{ij} = \log(d_{ij} + 1)$
  2. $d_{ij} = d_{ij} \log \frac{\sum_k 1}{\sum_k \delta_{ik}}$
  3. $d_{ij} = \frac{d_{ij}}{\sqrt{\sum_k (d_{kj})^2}}$
  4. $\hat{\theta}_{ci} = \frac{\sum_{j:y_j \neq c} d_{ij} + \alpha_i}{\sum_{j:y_j \neq c} \sum_k d_{kj} + \alpha}$
  5. $w_{ci} = \log \theta_{ci}$
  6. $w_{ci} = \frac{w_{ci}}{\sum_i w_{ci}}$
  7. Let $t = (t_1, \ldots, t_n)$ be a test document; let t be the count of word i.
  8. Label the document according to

$$l(t) = \arg\min_c \sum_i t_i w_{ci}$$

Our Results

Analysis - Balanced Classes

20NEWSGROUP Data with 4 Classes

Naive bayes	93.8%
Complement Naive bayes	93.2%

Accuracy =

(Correct predictions/Total Examples)*100

Analysis - UnBalanced Classes

20NEWSGROUP Data with 20 Classes

Naive bayes	78.71%
Complement Naive bayes	81.73%

Dataset Link :

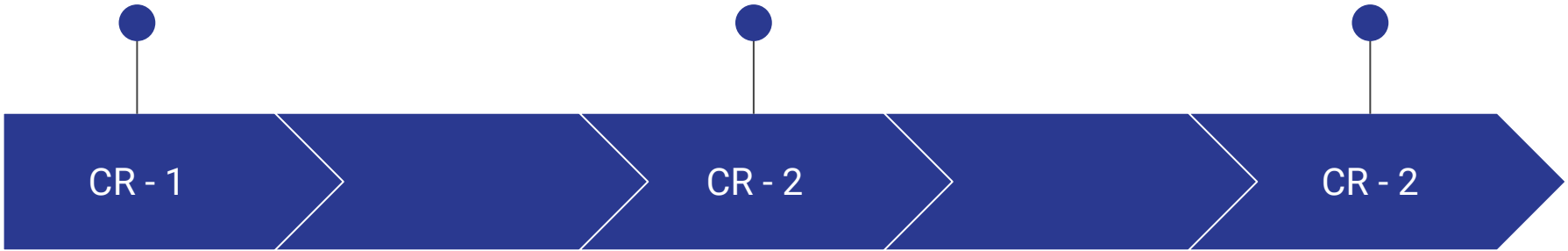
http://archive.ics.uci.edu/ml/datasets/twenty+newsgroups

Reviews

Still Lack to Captures Complete Dependencies of word in Document

CNB is Relatively is more Computational than classical NB

There are classifiers like SVM which don’t take independence assumption and performs better than NB



References

Tackling the Poor Assumptions of Naive Bayes Text Classifiers

Authors:Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, David R. Karger

Link: https://www.aaai.org/Papers/ICML/2003/ICML03-081.pdf

Thanks You!