Day 5: Classification and Scaling

Kenneth Benoit

Spring 2014

March 10, 2014
Today’s Road Map

Principles of “text as data” approaches
Introduction to the Naive Bayes Classifier
The k-Nearest Neighbour Classifier
“Wordscores” as an extension of Naive Bayes
Lab session: Classifying Text Using Wordstat
“TEXT AS DATA”
Scale this?
Pros and Cons of the “text as data” approach

- Fully automated technique with minimal human intervention or judgment calls – only with regard to reference text selection
- Language-blind
- (Pro) Inherits all the advantages of statistical data analysis
- (Con) very hard to understand the data-generating process
INTRODUCTION TO NAIVE BAYES
Prior probabilities and updating

A test is devised to automatically flag racist news stories.

- 1% of news stories in general have racist messages
- 80% of racist news stories will be flagged by the test
- 10% of non-racist stories will also be flagged

We run the test on a new news story, and it is flagged as racist.

Question: What is probability that the story is actually racist?

Any guesses?
Prior probabilities and updating

What about without the test?
- Imagine we run 1,000 news stories through the test
  - We expect that 10 will be racist

With the test, we expect:
- Of the 10 found to be racist, 8 should be flagged as racist
- Of the 990 non-racist stories, 99 will be wrongly flagged as racist
- That’s a total of 107 stories flagged as racist

So: the updated probability of a story being racist, conditional on being flagged as racist, is \( \frac{8}{107} = 0.075 \)
The prior probability of 0.01 is updated to only 0.075 by the positive test result
This is an example of Bayes’ Rule:

\[
P(R = 1 \mid T = 1) = \frac{P(T = 1 \mid R = 1)P(R = 1)}{P(T = 1)}
\]
Multinomial Bayes model of Class given a Word

Consider $J$ word types distributed across $I$ documents, each assigned one of $K$ classes.

At the word level, Bayes Theorem tells us that:

$$P(c_k|w_j) = \frac{P(w_j|c_k)P(c_k)}{P(w_j)}$$

For two classes, this can be expressed as

$$= \frac{P(w_j|c_k)P(c_k)}{P(w_j|c_k)P(c_k) + P(w_j|c_{-k})P(c_{-k})}$$ (1)
Classification as a goal

- Machine learning focuses on identifying classes (classification), while social science is typically interested in locating things on latent traits (scaling).
- One of the simplest and most robust classification methods is the “Naive Bayes” (NB) classifier, built on a Bayesian probability model.
- The class predictions for a collection of words from NB are great for classification, but useless for scaling.
- But intermediate steps from NB turn out to be excellent for scaling purposes, and identical to Laver, Benoit and Garry’s “Wordscores”.
- Applying lessons from machine to learning to supervised scaling, we can:
  - Apply classification methods to scaling
  - Improve it using lessons from machine learning.
Supervised v. unsupervised methods compared

- The goal (in text analysis) is to differentiate *documents* from one another, treating them as “bags of words”
- Different approaches:
  - *Supervised methods* require a *training set* that exemplify constrasting *classes*, identified by the researcher
  - *Unsupervised methods* scale documents based on patterns of similarity from the term-document matrix, without requiring a training step
- Relative **advantage** of supervised methods:
  You already know the dimension being scaled, because you set it in the training stage
- Relative **disadvantage** of supervised methods:
  You *must* already know the dimension being scaled, because you have to feed it good sample documents in the training stage
Supervised v. unsupervised methods: Examples

- General examples:
  - Supervised: Naive Bayes, k-Nearest Neighbor, Support Vector Machines (SVM)
  - Unsupervised: correspondence analysis, IRT models, factor analytic approaches

- Political science applications
  - Supervised: Wordscores (LBG 2003); SVMs (Yu, Kaufman and Diermeier 2008); Naive Bayes (Evans et al 2007)
  - Unsupervised “Wordfish” (Slapin and Proksch 2008); Correspondence analysis (Schonhardt-Bailey 2008); two-dimensional IRT (Monroe and Maeda 2004)
Focus today

- The focus today will be on Naive Bayes
- We will also cover the Laver, Benoit and Garry (2003) “Wordscores” scaling method
Consider $J$ word types distributed across $I$ documents, each assigned one of $K$ classes.

*At the word level*, Bayes Theorem tells us that:

$$P(c_k|w_j) = \frac{P(w_j|c_k)P(c_k)}{P(w_j)}$$

For two classes, this can be expressed as:

$$= \frac{P(w_j|c_k)P(c_k)}{P(w_j|c_k)P(c_k) + P(w_j|c_{-k})P(c_{-k})}$$

(2)
Moving to the document level

- The “Naive” Bayes model of a joint document-level class posterior assumes conditional independence, to multiply the word likelihoods from a “test” document, to produce:

\[ P(c|d) = P(c) \prod_j \frac{P(w_j|c)}{P(w_j)} \]

- This is why we call it “naive”: because it (wrongly) assumes:
  - conditional independence of word counts
  - positional independence of word counts
Multinomial Bayes model of Class given a Word

Class-conditional word likelihoods

\[ P(c_k|w_j) = \frac{P(w_j|c_k)P(c_k)}{P(w_j|c_k)P(c_k) + P(w_j|c_{\neg k})P(c_{\neg k})} \]

- The word likelihood within class
- The maximum likelihood estimate is simply the proportion of times that word \( j \) occurs in class \( k \), but it is more common to use Laplace smoothing by adding 1 to each observed count within class
Multinomial Bayes model of Class given a Word

Word probabilities

\[ P(c_k|w_j) = \frac{P(w_j|c_k)P(c_k)}{P(w_j)} \]

- This represents the word probability from the training corpus
- Usually uninteresting, since it is constant for the training data, but needed to compute posteriors on a probability scale
Multinomial Bayes model of Class given a Word

Class prior probabilities

\[ P(c_k | w_j) = \frac{P(w_j | c_k)P(c_k)}{P(w_j | c_k)P(c_k) + P(w_j | c_{\neg k})P(c_{\neg k})} \]

- This represents the class prior probability
- Machine learning typically takes this as the document frequency in the training set
- This approach is flawed for scaling, however, since we are scaling the latent class-ness of an unknown document, not predicting class — uniform priors are more appropriate
Multinomial Bayes model of Class given a Word
Class posterior probabilities

\[ P(c_k|w_j) = \frac{P(w_j|c_k)P(c_k)}{P(w_j|c_k)P(c_k) + P(w_j|c_{\neg k})P(c_{\neg k})} \]

- This represents the posterior probability of membership in class \( k \) for word \( j \)
- Under *certain conditions*, this is identical to what LBG (2003) called \( P_{wr} \)
- Under those conditions, the LBG “wordscore” is the linear difference between \( P(c_k|w_j) \) and \( P(c_{\neg k}|w_j) \)
Naive Bayes Classification Example

(From Manning, Raghavan and Schütze, *Introduction to Information Retrieval*)

13.2 Naive Bayes text classification

Table 13.1 Data for parameter estimation examples.

<table>
<thead>
<tr>
<th>docID</th>
<th>words in document</th>
<th>in (c = \text{China})?</th>
</tr>
</thead>
<tbody>
<tr>
<td>training set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Chinese Beijing Chinese</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>Chinese Chinese Shanghai</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>Chinese Macao</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>Tokyo Japan Chinese</td>
<td>no</td>
</tr>
<tr>
<td>test set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>
Example 13.1: For the example in Table 13.1, the multinomial parameters we need to classify the test document are the priors $\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$ and the following conditional probabilities:

\[
\begin{align*}
\hat{P}(\text{Chinese}|c) &= (5 + 1)/(8 + 6) = 6/14 = 3/7 \\
\hat{P}(\text{Tokyo}|c) &= \hat{P}(\text{Japan}|c) = (0 + 1)/(8 + 6) = 1/14 \\
\hat{P}(\text{Chinese}|\overline{c}) &= (1 + 1)/(3 + 6) = 2/9 \\
\hat{P}(\text{Tokyo}|\overline{c}) &= \hat{P}(\text{Japan}|\overline{c}) = (1 + 1)/(3 + 6) = 2/9
\end{align*}
\]

The denominators are $(8 + 6)$ and $(3 + 6)$ because the lengths of $text_c$ and $text_{\overline{c}}$ are 8 and 3, respectively, and because the constant $B$ in Equation (13.7) is 6 as the vocabulary consists of six terms.

We then get:

\[
\begin{align*}
\hat{P}(c|d_5) &\propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003. \\
\hat{P}(\overline{c}|d_5) &\propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001.
\end{align*}
\]

Thus, the classifier assigns the test document to $c = \text{China}$. The reason for this classification decision is that the three occurrences of the positive indicator Chinese in $d_5$ outweigh the occurrences of the two negative indicators Japan and Tokyo.
The class predictions for a collection of words from NB can be adapted to scaling.

The intermediate steps from NB turn out to be excellent for scaling purposes, and identical to Laver, Benoit and Garry’s “Wordscores”.

There are certain things from machine learning that ought to be adopted when classification methods are used for scaling:

- Feature selection
- Stemming/pre-processing
ALTERNATIVES: kNN
Other classification methods: \( k \)-nearest neighbour

- A non-parametric method for classifying objects based on the training examples that are closest in the feature space.

- A type of instance-based learning, or “lazy learning” where the function is only approximated locally and all computation is deferred until classification.

- An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its \( k \) nearest neighbors (where \( k \) is a positive integer, usually small).

- Extremely simple: the only parameter that adjusts is \( k \) (number of neighbors to be used) - increasing \( k \) smooths the decision boundary.
$k$-NN Example: Red or Blue?
$k = 1$
$k = 7$
$k = 15$
Distance usually relates to all the attributes and assumes all of them have the same effects on distance.

Misclassification may result from attributes not confirming to this assumption (sometimes called the “curse of dimensionality”) – solution is to reduce the dimensions.

There are (many!) different metrics of distance.
“WORDSCORES”
Wordscores conceptually

- Two sets of texts
  - Reference texts: texts about which we know something (a scalar dimensional score)
  - Virgin texts: texts about which we know nothing (but whose dimensional score we'd like to know)

- These are analogous to a “training set” and a “test set” in classification

- Basic procedure:
  1. Analyze reference texts to obtain word scores
  2. Use word scores to score virgin texts
Wordscores Procedure

The Wordscore Procedure
(Using the UK 1997-2001 Example)

Step 1: Obtain reference texts with a priori known positions (setref)
Step 2: Generate word scores from reference texts (wordscore)
Step 3: Score each virgin text using word scores (textscore)
Step 4: (optional) Transform virgin text scores to original metric
Wordscores Procedure

The Wordscore Procedure
(Using the UK 1997-2001 Example)

1. Labour 1992 5.35
2. Liberals 1992 8.21
3. Cons. 1992 17.21

Step 1: Obtain reference texts with a priori known positions ($setref$)
Step 2: Generate word scores from reference texts ($wordscore$)
Step 3: Score each virgin text using word scores ($textscore$)
Step 4: (optional) Transform virgin text scores to original metric

Scored word list:
- drugs 15.66
- corporation 15.66
- successfully 15.26
- markets 15.12
- motorway 14.96
- nation 12.44
- pensionable 11.59
- management 11.56
- monetary 10.84
- secure 10.44
- minorities 9.95
- women 8.65
- cooperation 8.64
- representation 7.42
- poverty 6.87
- waste 6.83
- unemployment 6.76
- contributions 6.68

Reference Texts

Scored virgin texts

Labour 1997 9.17 (.33)
Liberals 1997 5.00 (.36)
Cons. 1997 17.18 (.32)
The Wordscore Procedure
(Using the UK 1997-2001 Example)

Step 1: Obtain reference texts with a priori known positions (setref)
Step 2: Generate word scores from reference texts (wordscore)
Step 3: Score each virgin text using word scores (textscore)
Step 4: (optional) Transform virgin text scores to original metric
Wordscores Procedure

The Wordscore Procedure
(Using the UK 1997-2001 Example)

Step 1: Obtain reference texts with a priori known positions (setref)
Step 2: Generate word scores from reference texts (wordscore)
Step 3: Score each virgin text using word scores (textscore)
Step 4: (optional) Transform virgin text scores to original metric

Reference Texts

1. Labour 1992 5.35
2. Liberals 1992 8.21
3. Cons. 1992 17.21

Scored word list

- drugs 15.66
- corporation 15.66
- inheritance 15.48
- successfully 15.26
- markets 15.12
- motorway 14.96
- nation 12.44
- single 12.36
- pensionable 11.59
- management 11.56
- monetary 10.84
- secure 10.44
- minorities 9.95
- women 8.65
- cooperation 8.64
- transform 7.44
- representation 7.42
- poverty 6.87
- waste 6.83
- unemployment 6.76
- contributions 6.68

Scored virgin texts

1. Labour 1997 9.17 (.33)
2. Liberals 1997 5.00 (.36)
3. Cons. 1997 17.18 (.32)
Wordscores mathematically: Reference texts

- Start with a set of \( I \) reference texts, represented by an \( I \times J \) document-term frequency matrix \( C_{ij} \), where \( i \) indexes the document and \( j \) indexes the \( J \) total word types.

- Each text will have an associated “score” \( a_i \), which is a single number locating this text on a single dimension of difference.
  - This can be on a scale metric, such as 1–20.
  - Can use arbitrary endpoints, such as -1, 1.

- We normalize the document-term frequency matrix within each document by converting \( C_{ij} \) into a relative document-term frequency matrix (within document), by dividing \( C_{ij} \) by its word total marginals:

\[
F_{ij} = \frac{C_{ij}}{C_i}.
\]  

(3)

where \( C_i = \sum_{j=1}^{J} C_{ij} \).
Wordscores mathematically: Word scores

- Compute an $I \times J$ matrix of relative document probabilities $P_{ij}$ for each word in each reference text, as

$$P_{ij} = \frac{F_{ij}}{\sum_{i=1}^{I} F_{ij}}$$  \hspace{1cm} (4)

- This tells us the probability that given the observation of a specific word $j$, that we are reading a text of a certain reference document $i$
Assume we have two reference texts, A and B

The word “choice” is used 10 times per 1,000 words in Text A and 30 times per 1,000 words in Text B

So \( F_i \text{ "choice"} = \{0.010, 0.030\} \)

If we know only that we are reading the word choice in one of the two reference texts, then probability is 0.25 that we are reading Text A, and 0.75 that we are reading Text B

\[
P_A \text{ "choice"} = \frac{0.010}{(0.010 + 0.030)} = 0.25 \tag{5}
\]

\[
P_B \text{ "choice"} = \frac{0.030}{(0.010 + 0.030)} = 0.75 \tag{6}
\]
Wordscores mathematically: Word scores

Compute a $J$-length “score” vector $S$ for each word $j$ as the average of each document $i$’s scores $a_i$, weighted by each word’s $P_{ij}$:

$$S_j = \sum_{i=1}^{I} a_i P_{ij} \quad (7)$$

In matrix algebra, $S = a \cdot P$

This procedure will yield a single “score” for every word that reflects the balance of the scores of the reference documents, weighted by the relative document frequency of its normalized term frequency.
Continuing with our example:

- We “know” (from independent sources) that Reference Text A has a position of $-1.0$, and Reference Text B has a position of $+1.0$
- The score of the word choice is then 
  $0.25(-1.0) + 0.75(1.0) = -0.25 + 0.75 = +0.50$
Wordscores mathematically: Scoring “virgin” texts

- Here the objective is to obtain a single score for any new text, relative to the reference texts.
- We do this by taking the mean of the scores of its words, weighted by their term frequency.
- So the score $v_k$ of a virgin document $k$ consisting of the $j$ word types is:

$$v_k = \sum_j (F_{kj} \cdot s_j) \quad (8)$$

where $F_{kj} = \frac{C_{kj}}{C_k}$ as in the reference document relative word frequencies.

- Note that new words outside of the set $J$ may appear in the $K$ virgin documents — these are simply ignored (because we have no information on their scores).
- Note also that nothing prohibits reference documents from also being scored as virgin documents.
Wordscores mathematically: Rescaling raw text scores

- Because of overlapping or non-discriminating words, the raw text scores will be dragged to the interior of the reference scores (we will see this shortly in the results)
- Some procedures can be applied to rescale them, either to a unit normal metric or to a more “natural” metric
- Martin and Vanberg (2008) have proposed alternatives to the LBG (2003) rescaling
The score $v_k$ of any text represents a weighted mean.

LBG (2003) used this logic to develop a standard error of this mean using a *weighted variance* of the scores in the virgin text.

Given some assumptions about the scores being fixed (and the words being conditionally independent), this yields approximately normally distributed errors for each $v_k$.

An alternative would be to bootstrap the textual data prior to constructing $C_{ij}$ and $C_{kj}$ — see Lowe and Benoit (2012).
Suggestions for choosing reference texts

- Texts need to contain information representing a clearly dimensional position
- Dimension must be known a priori. Sources might include:
  - Survey scores or manifesto scores
  - Arbitrarily defined scales (e.g. -1.0 and 1.0)
- Should be as discriminating as possible: extreme texts on the dimension of interest, to provide reference anchors
- Need to be from the same lexical universe as virgin texts
- Should contain lots of words
Suggestions for choosing reference values

- Must be “known” through some trusted external source
- For any pair of reference values, all scores are simply linear rescalings, so might as well use (-1, 1)
- The “middle point” will not be the midpoint, however, since this will depend on the relative word frequency of the reference documents
- Reference texts if scored as virgin texts will have document scores more extreme than other virgin texts
- With three or more reference values, the mid-point is mapped onto a multi-dimensional simplex. The values now matter but only in relative terms (we are still investigating this fully)
Multinomial Bayes model of Class given a Word

Class posterior probabilities

\[ P(c_k | w_j) = \frac{P(w_j | c_k)P(c_k)}{P(w_j | c_k)P(c_k) + P(w_j | c_{\neg k})P(c_{\neg k})} \]

- This represents the posterior probability of membership in class \( k \) for word \( j \)
- Under *certain conditions*, this is identical to what LBG (2003) called \( P_{wr} \)
- Under those conditions, the LBG “wordscore” is the linear difference between \( P(c_k | w_j) \) and \( P(c_{\neg k} | w_j) \)
The LBG approach required the identification not only of texts for each training class, but also “reference” scores attached to each training class.

Consider two “reference” scores $s_1$ and $s_2$ attached to two classes $k = 1$ and $k = 2$. Taking $P_1$ as the posterior $P(k = 1|w = j)$ and $P_2$ as $P(k = 2|w = j)$, a generalised score $s_j^*$ for the word $j$ is then

$$s_j^* = s_1 P_1 + s_2 P_2$$
$$= s_1 P_1 + s_2 (1 - P_1)$$
$$= s_1 P_1 + s_2 - s_2 P_1$$
$$= P_1 (s_1 - s_2) + s_2$$
For more than two reference classes, if the reference scores are ordered such that $s_1 < s_2 < \cdots < s_K$, then

\[
s^*_j = s_1 P_1 + s_2 P_2 + \cdots + s_K P_K
\]

\[
= s_1 P_1 + s_2 P_2 + \cdots + s_K \left(1 - \sum_{k=1}^{K-1} P_k\right)
\]

\[
= \sum_{k=1}^{K-1} P_i(s_k - s_K) + s_l
\]
A simpler formulation:
Use reference scores such that $s_1 = -1.0$, $s_K = 1.0$

- From above equations, it should be clear that any set of reference scores can be linearly rescaled to endpoints of $-1.0, 1.0$
- This simplifies the “simple word score”

$$ s_j^* = (1 - 2P_1) + \sum_{k=2}^{K-1} P_k(s_k - 1) $$

- which simplifies with just two reference classes to:

$$ s_j^* = 1 - 2P_1 $$
Implications

- LBG’s “word scores” come from a linear combination of class posterior probabilities from a Bayesian model of class conditional on words
- We might as well always anchor reference scores at $-1.0, 1.0$
- There is a special role for reference classes in between $-1.0, 1.0$, as they balance between “pure” classes — more in a moment
- There are alternative scaling models, such that used in Beauchamp’s (2012) “Bayesscore”, which is simply the difference in logged class posteriors at the word level. For $s_1 = -1.0, s_2 = 1.0$,

$$s_j^B = -\log P_1 + \log P_2$$

$$= \log \frac{1 - P_1}{P_1}$$
The “Naive” Bayes model of a joint document-level class posterior assumes conditional independence, to multiply the word likelihoods from a “test” document, to produce:

\[ P(c|d) = P(c) \frac{\prod_j P(w_j|c)}{P(w_j)} \]

So we could consider a document-level relative score, e.g. 
1 – 2P(c_1|d) (for a two-class problem)

But this turns out to be useless, since the predictions of class are highly separated
- A better solution is to score a test document as the arithmetic mean of the scores of its words.
- This is exactly the solution proposed by LBG (2003).
- Beauchamp (2012) proposes a “Bayesscore” which is the arithmetic mean of the log difference word scores in a document – which yields extremely similar results.

And now for some demonstrations with data...
Application 1: Dail speeches from LBG (2003)

(a) NB Speech scores by party, smooth=0, imbalanced priors

(b) Document scores from NB v. Classic Wordscores

- three reference classes (Opposition, Opposition, Government) at \{-1, -1, 1\}
- no smoothing
Application 1: Dail speeches from LBG (2003)

(c) NB Speech scores by party, smooth=1, uniform class priors

(d) Document scores from NB v. Classic Wordscores

- two reference classes (Opposition + Opposition, Government) at \{-1, 1\}
- Laplace smoothing
Application 2: Classifying legal briefs (Evans et al 2007)

Wordscores v. Bayesscore

(a) Word level

(b) Document level

- Training set: Petitioner and Respondent litigant briefs from *Grutter/Gratz v. Bollinger* (a U.S. Supreme Court case)
- Test set: 98 amicus curiae briefs (whose P or R class is known)
Application 2: Classifying legal briefs (Evans et al 2007)
Posterior class prediction from NB versus log wordscores

Log wordscores mean for document
Posterior $P(\text{class}=\text{Petitioner}|\text{document})$
Predicted Petitioner
Predicted Respondent

Predicted Petitioner
Predicted Respondent
Application 3: LBG’s British manifestos
More than two reference classes

- x-axis: Reference scores of \{5.35, 8.21, 17.21\} for Lab, LD, Conservatives
- y-axis: Reference scores of \{10.21, 5.26, 15.61\}
Application 4: Back to Evans et al (2007) for some Feature Selection: Classification results

<table>
<thead>
<tr>
<th>Smoothing</th>
<th>Stopwords</th>
<th>Bigrams</th>
<th>Distribution</th>
<th>Wordscores Accuracy</th>
<th>F1</th>
<th>Naive Bayes Scaling Accuracy</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Multi</td>
<td>0.897</td>
<td>0.836</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Bern</td>
<td>0.459</td>
<td>0.647</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Add-1</td>
<td>No</td>
<td>No</td>
<td>Multi</td>
<td>0.897</td>
<td>0.836</td>
<td>0.897</td>
<td>0.836</td>
</tr>
<tr>
<td>Add-1</td>
<td>No</td>
<td>No</td>
<td>Bern</td>
<td>-</td>
<td>-</td>
<td>0.489</td>
<td>0.63</td>
</tr>
<tr>
<td>Add-1</td>
<td>Yes</td>
<td>No</td>
<td>Multi</td>
<td>0.897</td>
<td>0.843</td>
<td>0.918</td>
<td>0.86</td>
</tr>
<tr>
<td>Add-1</td>
<td>Yes</td>
<td>No</td>
<td>Bern</td>
<td>-</td>
<td>-</td>
<td>0.500</td>
<td>0.62</td>
</tr>
<tr>
<td>Add-1</td>
<td>Yes</td>
<td>Yes</td>
<td>Multi</td>
<td>0.887</td>
<td>0.810</td>
<td>0.897</td>
<td>0.836</td>
</tr>
<tr>
<td>Add-1</td>
<td>Yes</td>
<td>Yes</td>
<td>Bern</td>
<td>-</td>
<td>-</td>
<td>0.785</td>
<td>0.71</td>
</tr>
</tbody>
</table>

Relative performance of NB and Wordscores as classifiers, given different feature selection.

*(F1 score is the harmonic mean of average precision and recall)*
Conclusions

- The venerable LBG 2003 wordscores method is based on an underlying Bayesian probability model.
- Naive Bayes class prediction is useless for scaling, but Bayesian posterior scaling (using arithmetic means) is (also) useful for classification.
- Always use $-1, 1$ reference scores.
- Two class training sets are preferred, since middle classes only combine extreme classes.
- Use uniform priors – this implies aggregating training documents by class.
- No knockout results from feature selection so far, implying just using the unfiltered texts seems to be OK for supervised methods.