Journal of Artificial Intelligence Research vol (year) pages Submitted submission-date; published publication-date

Significance Testing against the Random Model for Scoring Models on Top k Predictions[0]

Sofus A. Macskassy SMACSKAS@STERN.NYU.EDU
Information, Operations and Management Sciences
NYU Stern School of Business, 44 W. 4th St, New York, NY 10012

Abstract

Performance at top k predictions, where instances are ranked by a (learned) scoring model, has been used as an evaluation metric in machine learning for various reasons such as where the entire corpus is unknown (*e.g.*, the web) or where the results are to be used by a person with limited time or resources (*e.g.*, ranking financial news stories where the investor only has time to look at relatively few stories per day). This evaluation metric is primarily used to report whether the performance of a given method is significantly better than other (baseline) methods. It has not, however, been used to show whether the result is *significant* when compared to the simplest of baselines — the random model. If no models outperform the random model at a given confidence interval, then the results may not be worth reporting. This paper introduces a technique to perform an analysis of the expected performance of the top k predictions from the random model given k and a p-value on an evaluation dataset $\mathcal{D}$. The technique is based on the realization that the distribution of the number of positives seen in the top k predictions follows a *hypergeometric distribution*, which has well-defined statistical density functions. As this distribution is discrete, we show that using parametric estimations based on a binomial distribution are almost always in complete agreement with the discrete distribution and that, if they differ, an interpolation of the discrete bounds gets very close to the parametric estimations. The technique is demonstrated on results from three prior published works, in which it clearly shows that even though performance is greatly increased (sometimes over 100%) with respect to the expected performance of the random model (at $p = 0.5$), these results, although qualitatively impressive, are not always as significant ($p = 0.1$) as might be suggested by the impressive qualitative improvements. The technique is used to show, given k, both how many positive instances are needed to achieve a specific significance threshold is as well as how significant a given top k performance is. The technique when used in a more global setting is able to identify the crossover points, with respect to k, when a method becomes significant for a given p. Lastly, the technique is used to generate a complete confidence curve, which shows a general trend over all k and visually shows where a method is significantly better than the random model over all values of k.

1. Introduction

Consider a web search engine or a personalized news portal. In the former case, searching the web often results in thousands or millions page hits. In the latter case, the site would give you the news stories that fit your profile best. In both cases, what are generally looked at, and hence of interest, are the highest ranked instances. How would one evaluate or measure the value of the information presented by the search engine or portal? A common measure is precision at top k predictions, where the most commonly used values of k are $5, 10, 20, 30, 100$ (Dumais & Nielsen, 1992; Mitra,

0. S.A. Macskassy, "Significance Testing against the Random Model for Scoring Models on Top k Predictions" CeDER Working Paper CeDER-05-09, Stern School of Business, New York University, NY, NY 10012. January 2005.

Singhal, & Buckley, 1998; Cohen, Schapire, & Singer, 1999; Singhal, Abney, Bacchiani, Collins, Hindle, & Pereira, 1999; Basu, Hirsh, Cohen, & Nevill-Manning, 2001).

More generally, for this paper, we assume a binary classification task, where the problem addressed is that of evaluating a (learned) scoring model:

$$f : X \rightarrow \mathcal{R}, \tag{1}$$

where $f(\cdot)$ returns a score for x which can then be used to rank instances. Higher scores indicate that x should be ranked higher than an x with a lower score because it is more likely to be of the positive class. The task is to evaluate $f(\cdot)$ based on its performance given its top k predictions—*e.g.*, the k highest scoring instances x_i in a given data set $\mathcal{D}$. Often, the top k comparison metric has been used to compare two learned models to see which is the better performer, where "better" is defined as having more true positives in the top k predictions—the topic of this paper. This metric is generally converted into precision or accuracy at top k ($P@k$), which is what is generally reported. However, the basic question that is rarely asked is whether the performance is better than what would be expected from the *random model*—a model which returns a random ordering of $\mathcal{D}$ assuming all orderings are equally likely. It might well be that a given model is significantly better than another, but if neither performs better than what could be expected from the random model, then the relative performance of the models are not that interesting.

The main contribution of this paper is the introduction of an efficient technique for generating a confidence bound based on k in $O(k^2)$ time. An extra boon as that it generates all the bounds for $1, \ldots, k$ in the process. Given a p-value and an evaluation data set $\mathcal{D}$ where the labels are known, the confidence bound will define how many positives are needed in the top k predictions in order to outperform a random ranking with a confidence of $(1-p)$. The technique needs only two data characteristic parameters, sample size (N) and the number of positives (N^+), to generate a confidence bound for a given p.[1] In the case of $\mathcal{D}$, these parameters would be:

$$N = |\mathcal{D}| \tag{2}$$
$$N^+ = |\mathcal{D}^+|, \tag{3}$$

where

$$\mathcal{D}^+ = \{x_i | x_i \in D, \text{label}(x_i) = +\}. \tag{4}$$

Knowing N^+ (and N^-) makes it possible to know exactly how many positives and negatives are left after seeing the top k predictions (and observing how many of those were positives.) This fact is a key in how the algorithm is derived and allows the bounds to be tailored to the characteristics of the given data set as opposed to the more general case of infinite data.

One straightforward way to calculate this bound is by considering all possible orderings, or rankings, of $\mathcal{D}$ and picking the number of positives in top k needed such that $(1-p)$ of all possible rankings have at most that number of positives in their top k ranking. For example, to get the random bound at $k = 5$ with $p = 0.05$, you would need to compute the number of positives needed in the top 5 predictions in order to have at least as many positives as would be seen in 95% of all rankings. Although it is trivial to examine all rankings of $\mathcal{D}$ when $\mathcal{D}$ is small, it quickly becomes intractable to do examine the $N!$ possible rankings as $\mathcal{D}$ grows. The technique used here overcomes

1. We will later show how the requirement of knowing $\mathcal{D}$ can be loosened to work in the infinite case as long as the class distribution is known.

this obstacle by first showing how the bounds can be calculated for all k in $O(N^3)$ time, and then gives an efficient algorithm which calculates the positives needed at top k, for $k = 1, \ldots, N$ in $O(N^+ \cdot N^-)$ time and $O(N)$ space.

The rest of this paper is outlined as follows: Section 2 describes the technique for generating the top k confidence bounds, Section 3 demonstrates the use of this technique on published prior results, and is followed by final remarks.

2. Top k Confidence Bounds for the Random Model

This section first describes the theory behind generating a confidence bound for the expected performance of the random model at top k predictions, then describes an $O(N^+ \cdot N^-)$ implementation to generate the confidence bound for all $k = 1, \ldots, N$.

2.1 Generating the Top k Confidence Bound

Let $\mathbf{R}$ denote the set of all rankings of $\mathcal{D}$ and let $q_{ik} \in \mathbf{q}_k$ be the number of positives seen in the top k instances of $R_i \in \mathbf{R}$. Given p and k, we can compute $n(k,p)$, the value which is the i^{th} largest in $\mathbf{q}_k$, where $i = p\|\mathbf{q}_k\|$. That is the value which represents the number of positives needed in the top k predictions in order to dominate $(1-p)$ of all the rankings $R_i \in \mathbf{R}$. However, computing this number would require sorting $\mathbf{q}_k$, which contains $N!$ values and is therefore not tractable for large data sets.

If we knew the distribution of values for $\mathbf{q}_k$ then we could use the distribution directly to compute $n(k,p)$ by using the cumulative density function (cdf). Specifically, $n(k,p)$ would be the value of i at which the cdf equals $(1-p)$. We observe that $\mathbf{q}_k$ is a *hypergeometric distribution* (Weisstein, 2002) with parameters N for population size, N^+ for success population size and k for the sample size. The distribution has the following statistical properties:

$$P(q_k = i) = \frac{\binom{N^+}{i}\binom{N^-}{k-i}}{\binom{N}{k}} \tag{5}$$

$$\text{cdf}(i,k) = \begin{cases} 0 & \text{for } i < \max(0, k + N^+ - N), \\ \sum_{j=\max(0,k+N^+-N)}^{\lfloor i \rfloor} P(q_k = j) & \text{for } 0 \le i < \min(k, N^+), \\ 1 & \text{for } i \ge \min(k, N^+) \end{cases} \tag{6}$$

where $\text{cdf}(i,k)$ is the cumulative density function representing $P(q_k \le i|\mathbf{q}_k)$. Using these, calculating $n(k,p)$ is straightforward:

$$n(k,p) = \arg\min_i \left[\text{cdf}(i,k) > 1 - p\right] \tag{7}$$

Note that the hypergeometric distribution is a discrete distribution. Therefore, $P(q_k = i)$ is defined only over discrete values of i and $n(k,p)$ will necessarily be an integer. This creates a possible problem when evaluating a given model. Often reported results are based on averaged runs and are therefore not discrete but continuous. Further, $n(k,p)$ can be the same for a range of p—for example when $k = 1$, then $n(k,p) = 1$ for any p-value less than $\frac{N^-}{N}$. A large range makes it harder to generate fine-grained confidence bounds. In such cases, one would have to either (1) truncate the

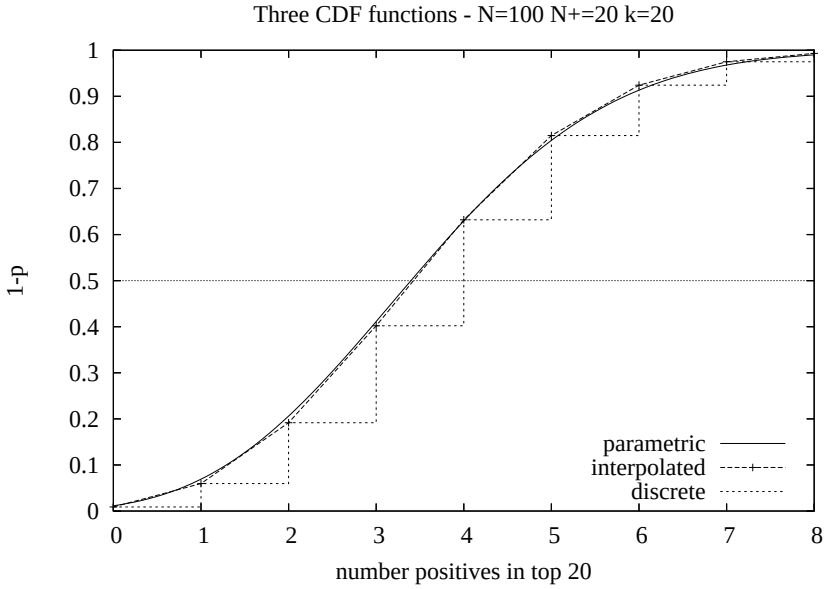

Figure 1: Example cdf curves for $N = 200$, $N^+ = 40$ at $k = 20$. The horizontal line represents $p = 0.5$.

averaged result, n, (as seen in Equation 6) when computing $n(k, p)$ equivalently report the smallest p'-value, $p' > p$, such that $n(k, p')$ is defined, (2) interpolate between the two discrete values that straddle n, or (3) use parametric estimations of $n(k, p)$ and p.

Fortunately, it is possible to estimate p (and n) by taking advantage of the fact that the hypergeometric distribution is very similar to the binomial distribution. In fact, the two distributions converge as $N \to \infty$. Therefore, it is possible to estimate $n(k, p)$ for any continuous p as well as p for any given continuous n using the following parametric estimation of the binomial cdf:

$$\text{cdf}_{\text{bin}}(i, k) = I_z(i + 1, N - i) \equiv \frac{B(z; i + 1, N - i)}{B(i + 1, N - i)}, \tag{8}$$

where $z = \frac{N^+}{N}$ and $B(z; a, b)$ is the incomplete beta function:

$$B(z; a, b) = \int_0^z u^{(a-1)}(1 - u)^{(b-1)} du \tag{9}$$

and $B(a, b)$ is the complete beta function ($B(a, b) = B(1; a, b)$, which for our domain always equals 1 and can therefore be ignored.) Estimating $B(p; a, b)$ is obviously much more computationally intensive than using the discrete distribution. Not only is it more efficient to use the discrete values but, as we shall see, using interpolation gets very close to the parametric estimations and ought therefore to be used for efficiency reasons. Figure 1 shows the three cdfs under consideration and where they cross the p-value line at $(1 - p) = 0.5$.[2] The figure clearly shows the problem when

2. We use gnuplot's `ibeta` function to compute the parametric cdf and the parametric bounds in the examples below.

x is real, where the discrete cdf would always round up (it crosses the line at $x = 4$). Second, we see that the interpolated and parametric curves both follow each other very closely although the parametric curve is not always in agreement even at the discrete points. As N increases, however, these do converge and the parametric cdf ends up being virtually the same as the discrete cdf at the discrete points. In the figure, the interpolated cdf intersects the given p-value line at $x = 3.43$ where the parametric curve crosses at $x = 3.40$. We also see that the interpolated curve does not always fall above or below the parametric curve due to the parametric curve being off at the discrete points. Therefore, for small data sets, we cannot generally say whether the interpolated bound will be larger or smaller than the parametric bound.

Computing the discrete version of $n(k, p)$ for a given k is relatively inexpensive as computing $\binom{N}{k}$ can be done in $O(k)$ time. Therefore, calculating $\binom{N^+}{i} \cdot \binom{N^-}{k-i}$ for $i = 0, \ldots, k$ takes, for each k:

$$\sum_{i=0}^{k} O(i) + O(k-i) = O(k^2) \tag{10}$$

However, as we would need this for all k, we need to compute $n(k, p)$ for $k = 1, \ldots, N$:

$$\sum_{k=1}^{N} O(k^2) = O(N^3) \tag{11}$$

This can be optimized to run in $O(N^+ \cdot N^-)$ as the next section will show.

2.2 An Efficient Implementation to Calculate $n(k, p)$ for all k

The efficient implementation works by calculating $n(k, p)$, one k at a time, for $k = 1, \ldots, N$. Let $\mathbf{x}_i^k$ be the set of rankings which have i positives in their top k predictions. The implementation is based on the following observation: the ratio of rankings in $\mathbf{x}_i^k$ which also belong to $\mathbf{x}_{(i+1)}^{(k+1)}$ is exactly $\left(\frac{N^+-i}{N-k}\right)$, the ratio of positives that are left in the remaining $N - k$ instances. The same holds for the negative instances. The key additional observation is that $\mathbf{x}_{(i+1)}^{(k+1)}$ contains not only the rankings in $\mathbf{x}_i^k$ which have a positive in their $k + 1^{\text{st}}$ prediction, but also the rankings in $\mathbf{x}_{(i+1)}^k$ which do *not* have a positive in their $k + 1^{\text{st}}$ prediction. Both would result in rankings which have $i + 1$ positives in their $k + 1$ top-ranked instances.

For pragmatic reasons, the implementation uses the fractions, $r_i^k = \frac{|x_i^k|}{N!} = P(x_k = i)$, rather than manipulate the large values of $|x_i^k|$. The above observations hold for r_i^k as well, and are expressed more formally by the following:

$$\mathrm{P}\left(+|r_i^k\right) = \frac{N^+ - i}{N - k} \tag{12}$$

$$\mathrm{P}\left(-|r_i^k\right) = \frac{N^- - (k - i)}{N - k} \tag{13}$$

$$r_0^0 = 1 \tag{14}$$

$$r_i^k = \begin{cases} \mathrm{P}\left(+|r_{(i-1)}^{(k-1)}\right) \cdot r_{(i-1)}^{(k-1)} + \mathrm{P}\left(-|r_{(i)}^{(k-1)}\right) \cdot r_{(i)}^{(k-1)} & 0 < i < k \\ \mathrm{P}\left(-|r_{(i)}^{(k-1)}\right) \cdot r_{(i)}^{(k-1)} & i = 0 \\ \mathrm{P}\left(+|r_{(i-1)}^{(k-1)}\right) \cdot r_{(i-1)}^{(k-1)} & i = k \end{cases}, \tag{15}$$

input: N^+, N, p

begin:

$N^- \leftarrow (N - N^+)$
$\mathbf{N}^p \leftarrow []$
$t[0][0] \leftarrow 1$
for $k = 1, \ldots, N$
  $n^p \leftarrow 0$
  $\min_n \leftarrow \max(0, k - N^-)$
  $\max_n \leftarrow \min(N^+, k)$
  for $n = \min_n, \ldots, \max_n$
    $\text{parent}_l = \begin{cases} \left(\frac{N^+ - n}{N - k}\right) \cdot t[k-1][n-1] & n > 0 \\ 0 & \text{else} \end{cases}$
    $\text{parent}_r = \begin{cases} \left(\frac{N^- - (k-n)}{N-k}\right) \cdot t[k-1][n]) & p < k \\ 0 & \text{else} \end{cases}$
    $t[k][n] \leftarrow \text{parent}_l + \text{parent}_r$
    $n^p \leftarrow n^p + t[k][n]$
    if$(n^p > p)$ **and** $(\mathbf{N}^p[k] =)$ **then**
      $\mathbf{N}^p[k] = n$
    endif
  endfor
endfor

end

output: $\mathbf{N^p}$

Table 1: Pseudo code to generate $\mathbf{N}^p = \{n(k,p) | k = 1, \ldots, N\}$.

where P $\left(+|r_i^k\right)$ (P $\left(-|r_i^k\right)$) is the probability of having the $k+1^{\text{st}}$ prediction be positive (negative). The two boundary cases, r_k^k and r_0^k clearly depend only on $r_{(k-1)}^{(k-1)}$ and $r_0^{(k-1)}$, respectively. It is now possible to reformulate Equation 6 by replacing $P(x_k = j)$ with r_j^k:

$$\text{cdf}(i,k) = \begin{cases} 0 & \text{for } i < \max(0, k + N^+ - N), \\ \sum_{j=\max(0,k+N^+-N)}^{\lfloor i \rfloor} r_j^k & \text{for } 0 \leq i < \min(k, N^+), \\ 1 & \text{for } i \geq \min(k, N^+) \end{cases} \tag{16}$$

Table 1 shows an $O(N^+ \cdot N^-)$ algorithm based on these observations. Note that since the computation of $r[k][\cdot]$ only requires the values of $r[k-1][\cdot]$, we only need to keep $r[k][\cdot]$ and $r[k-1][\cdot]$ in memory at any given time thereby keeping space usage to $O(N)$.

Finally, note that the algorithm can easily be extended to handle the infinite data case by redefining $\mathrm{P}\left(+|r_i^k\right)$ and $\mathrm{P}\left(-|r_i^k\right)$:

$$\mathrm{P}_\infty\left(+|r_i^k\right) = \mathrm{P}(+) \quad (17)$$

$$\mathrm{P}_\infty\left(-|r_i^k\right) = (1-\mathrm{P}(+))\,. \quad (18)$$

In this case the algorithm would only need to know the class marginals and then be told the maximum k to generate its bounds for. As observed earlier, this will generate the binomial distribution.

3. Demonstration

This section demonstrates the use of top-k confidence bounds to evaluate whether a model is performing better than what would be expected from the random model.

3.1 Data and Prior Results

The demonstration uses results from 3 prior studies that reported enough information (data set size, number of positives and an evaluation using top k as a measure) to make it possible to assess significance. While there is much research in information retrieval and text retrieval, such as at the Text REtrieval Conference (TREC)[3], where this technique could be useful, often $P(+)$ is not known or the class skew is so large (< 100 out of 1 million) that seeing even one positive in the top 10 is significant at $p = 0.001$.

The first data set comes from the "Stock Movement" problem in our prior work (Macskassy, Hirsh, Provost, Sankaranarayanan, & Dhar, 2001). The data consists of $31,406$ news stories collected from newswires between January 5, 1999 and September 14, 1999. The data contains 6222 positive stories, where a positive story was identified as

> A news story is positive (interesting) if the stock price of a company mentioned in the story moves significantly in the hour following the story,

where a "significant movement" of a stock was defined as more than one standard deviation from the normal one-hour movement of that stock. This test set included $16,769$ stories, 3123 of which were labeled as positive (interesting) [$P(+) = 0.186$]. The evaluation measure used in this work was the number of positives at top $5, 10, 20, 100$ and their relative improvements over the expected performance (where the number of positives are expected to be $k \cdot P(+)$.) This is equivalent to calculating the significance bound at $p = 0.5$. Results were reported for three commonly used text classification systems: Rocchio (Joachims, 1997; Schapire, Singer, & Singhal, 1998; Sebastiani, 2002), Naive Bayes (Domingos & Pazzani, 1996; Joachims, 1997; Mitchell, 1997) and Maximum Entropy (McCallum & Nigam, 1998; Nigam, Lafferty, & McCallum, 1999).[4] The performance reported in the paper ranged from 0 to 4 positives in the top 5 (versus 1 positive expected), 3 to 5 positives in the top 10 (versus 2 expected), 6 to 9 in top 20 (verses 4 expected), and 32 to 45 in the top 100 predictions (versus an expected 19 positives). The improvements are clearly impressive, but are they significant?

3. `http://trec.nist.gov`
4. Using the RAINBOW text-classification system version 20020213 (McCallum, 1996).

The second set of results are based on the task of assigning submitted papers to reviewers (Dumais & Nielsen, 1992). The technique used was Latent Semantic Indexing (LSI) (Deerwester, Dumais, Landauer, Furnas, & Harshman, 1990), using various reduced dimension sizes. The relevant evaluation measure reported was precision at top 10 ($P@10$). We consider the data set consisting of 117 papers submitted to the ACM Hypertext 1991 conference. The data contains full feedback on relevance from 15 reviewers, where the ratings ranged from not relevant (to the reviewer's expertise) to very relevant. The mean number of relevant papers was 47 (giving $P@10 = 0.40$).[5] The reported result on this data was $P@10 = 0.57$. This is equivalent to having 5.7 positives in the top 10 predictions. The second result reported in this paper was across 10 benchmark data sets, which had a total number of 8629 documents. The performance reported was $P@10 = 0.59$, or 5.9 positives in the top 10 predictions, again with an expected $P@10 = 0.40$.

The third set of results is also based on assigning technical papers to reviewers (Basu et al., 2001). This data set consists of 256 papers submitted to AAAI-1998, again with feedback on relevance from the reviewers. The technique used was an RDBMS system called WHIRL (Cohen, 1998), which has the added functionality of being able to do joins based on text similarity rather than exact matching. The methodology used here was to generate a body of text that profiles a reviewer and to match it against a body of text representing a paper to see if there is a good fit. On average, reviewers found 18 papers to be relevant ($P(+) = 0.07$). The relevant evaluation measure used was $P@10$ and $P@30$. Various profiling representations were used such as whether to use a reviewer's home page, published papers or both. To predict whether to assign a paper to a reviewer, this profile was either matched to the paper's abstract, title, keyword or any possible combination thereof. The values of $P@10$ ranged from 0.210 to 308 and the values of $P@30$ ranged from 0.169 to 0.217.

3.2 Comparison to Random Confidence Bounds for Specific k

While the performances for the stock movement problem are clearly well above the expected performance using the random model at $p = 0.5$, are they *significantly* above what could be expected from the random model? Using the "stock-movement" test set as the data set from which to generate the random confidence bounds results in $\mathcal{D}$ consisting of $N = 16,769$ instances with $N^+ = 3123$ of those being positive.

Freezing k and p to particular values, it is now possible to ask whether the reported results are indeed significant. Specifically, the study reported values for $k \in \{5, 10, 20, 100\}$. We will consider $p \in \{0.100, 0.001\}$. To evaluate the significance of the reported results, two things need be considered:

1. Given p and k, how many positives would be expected? To answer this, we freeze p and k, and solve for j in Equation 7.

2. How significant would it be to see j positives in top k predictions? This is answered by freezing i and k, and computing $p = 1 - \text{cdf}(k, i)$ using the cdf function given in Equation 16.

The algorithm shown in Table 1 can easily be modified either (a) to return the number of expected positives, given k and p, or (b) to return the p-value for a given k and n^+, the number of observed positives in those top k. Table 2 shows n^+, the number of positive instances in each of the

5. One global threshold on the reviewer relevance rating was used to define 'relevant.'

	Prior	$n(x_k, p)$		Rocchio		Maximum Entropy		Naive Bayes	
k		$p=0.100$	$p=0.001$	n+	p	n+	p	n+	p
5	0.93	2/ 1.72/ 1.58	4/ 3.84/ 3.57	2	0.048/ 0.048	0	0.643/ 0.643	4	$2.2e^{-4}/2.2e^{-4}$
10	1.86	3/ 2.99/ 2.98	6/ 5.88/ 5.73	5	0.004/ 0.004	3	0.098/ 0.098	4	0.025/ 0.025
20	3.72	6/ 5.59/ 5.50	10/ 9.40/ 9.25	8	0.006/ 0.006	9	0.001/ 0.001	6	0.063/ 0.063
100	18.62	24/23.18/23.17	31/30.91/30.92	32	$4.1e^{-4}/4.3e^{-4}$	45	$3.3e^{-10}/3.8e^{-10}$	39	$5.1e^{-7}/5.5e^{-7}$

Table 2: Evaluation of performance on the "stock-movement" data for $k = \{5, 10, 20, 100\}$. Shown are the top k bounds. The first column shows the expected performance at $p = 0.5$ (the prior). The next two groupings show the expected performance (discrete/interpolated/parametric) at $p = 0.100$ and $p = 0.001$. The next three groupings show the reported number of positives for each k for each method (n^+) along with the computed p-value (discrete/parametric).

top k brackets, for $k \in \{5, 10, 20, 100\}$. Three possible bounds were generated for each p-value, either the discrete bound (finding the smallest integer whose cdf is larger than $(1-p)$), interpolation between the discrete bounds straddling $(1-p)$, and finally the parametric bound. Two things are apparent from the numbers in the table. The first is that although the methods generally were able to outperform the random model at $p = 0.5$, often by a large margin, the improvements were not as significant as they qualitatively look until k becomes large. The second noteworthy observation is that the computed p-values for the discrete and parametric bounds are almost always equal as seen in the last three groupings. This is due to having discrete values for n^+, where the parametric and discrete cdfs generally should be the closest as observed in Figure 1. For large data sets as this, the binomial estimation of $n(k, p)$ will be slightly lower than that of the discrete bound, but larger than the interpolated bound. This is due to the curvature of the cdf at the p-values we generally are interested in. Figure 2 highlights the bounds generated by the parametric and interpolated methods for $p = 0.1$ and $p = 0.01$, as well as the curves used to generate these bounds. As we can clearly see, the interpolated curve only touches the parametric curve at the discrete points and is otherwise below it.[6] Therefore, it will cross the p-value line for any given p later (with respect to n^+) than the parametric curve. This fact should lead one to conclude that using parametric bounds will yield better bounds when the values whose p-values are to be estimated are real. However, since the interpolated bounds are very close to the parametric bounds and generally are slightly more conservative, then you should consider using them as they are computed more efficiently than the parametric bounds. Further, the parametric bounds are not precise for smaller data sets as we shall see below.

Looking at the numbers from the discrete distribution, we see that only Naive Bayes is significantly better than the random model at $k = 5$. However, it then becomes less significant at $k = 10, 20$ until starting to improve again. Rocchio, on the other hand, is better at $k = 10$ with $p = 0.025$, but it has a slower rate of improvement as kv increases than that of Maximum Entropy, which is the best at $k = 20$ with $p = 0.006$. All of them do perform significantly better ($p < 0.001$) than the random model at $k = 100$.

6. This is always the case for larger data sets, although it does not happen for smaller data stes as shown in Figure 1 and as we will see below.

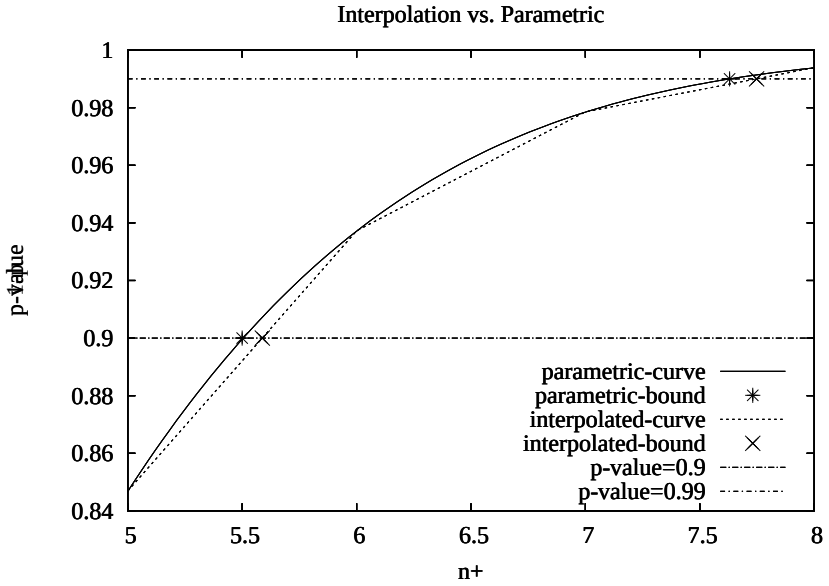

Figure 2: Figure showing why parametric bounds will generally be smaller than interpolated bounds.

	Size	Prior	$n(x_{10}, p)$		Reported	Computed
		$(p{=}0.5)$	$p{=}0.100$	$p{=}0.001$	$P@10$	p-value
Submitted Abstracts	117	4.0	6/5.09/5.50	8/7.88/8.22	5.7	0.166/0.088/0.079
All Datasets	8269	4.0	6/5.59/5.50	9/8.42/8.22	5.9	0.166/0.066/0.062

Table 3: Result for review assignments on Hypertext'91 and benchmark datasets (Dumais & Nielsen, 1992). The table shows the top k bounds, based on prior, expected performance with $p = 0.100$ and $p = 0.001$ (discrete/interpolated/parametric), and $P@10$ results as reported in the prior study with their computed p-values (discrete/interpolated/parametric).

	Evaluation Measure	Size	Prior (p=0.5)	$n(x_k, p)$ p=0.100	p=0.001	Reported num+	Computed p-value
Worst (h+A)	$P@10$	256	0.70	2/1.40/1.28	4/3.69/3.54	2.10	0.026/0.023/0.024
Worst (h+A)	$P@30$	256	2.10	4/3.45/3.45	7/6.56/6.89	5.07	0.011/0.010/0.015
Best (h+KT)	$P@10$	256	0.70	2/1.40/1.28	4/3.69/3.54	3.08	0.003/0.003/0.003
Best (h+K)	$P@30$	256	2.10	4/3.45/3.45	7/6.56/6.89	6.51	0.002/0.001/0.002

Table 4: Results for paper assignment prediction for AAAI-1998 submissions (Basu et al., 2001). It shows the number of positives in top k, based on prior, expected performance (discrete/interpolated/parametric) with $p = 0.100$ and $p = 0.001$, and results as reported in the prior study along with the computed p-values (discrete/interpolation/parametric).

We next analyze the reported performance of the LSI technique on the Hypertext'91 data and on the 10 benchmark data sets. Table 3 shows the results of the analysis. Although the reported results show an increase of the relevant documents in the top 10 predictions from 4 to 5.9 (which is truncated to 5 for the discrete case as is called for in Equation 6), this improvement is not as significant as one might have expected with $p = 0.081$ and $p = 0.059$ for the two interpolated bounds. The table shows clearly that it is possible to get very close to the parametric bound by using interpolation, whereas using the discrete bounds are off by a large margin. This argues that if one were to use the discrete bounds, then interpolation makes more sense than truncating the values. We again see that the parametric bound, when computing $n(x_k, p)$ for a given value of p, contains is on the order of 0.5 fewer positives in the top k than using the discrete bound.

The last reported results we analyze is that of predicting reviewer assignment fitness to 256 of the papers submitted to AAAI-1998 (Basu et al., 2001). Table 4 shows the significance of the best and worst results reported, where the worst reported results for both $P@10$ and $P@30$ used the homepage as a profile for a reviewer and matched it against the abstract of the paper. For the best results, both again used the homepage as the profile of the reviewer which was then matched against either the keywords and title of the paper ($P@10$) or only the keywords ($P@30$). The table shows that even the worst results for $P@10$ were fairly significant with $p = 0.023$, where the worst results for $P@30$ were even more significant at $p = 0.010$. The best results were highly significant for both values of k, again with a higher significance as k increased. Again, we see very close agreement between the bounds found using the parametric functions and the interpolated method. Interestingly, we see that for the two $P@30$ results, the p values were slightly lower for the discrete distribution over that of the parametric method. Performing the interpolation made this difference even larger. We plotted the two cdf curves as before to investigate this behavior. Figure 3 highlights the relevant part of the $P@30$ curves. Two noteworthy things happen. First, we see that the integration falls apart in two places for small values of n^+. Second, we see that the interpolated curve actually is partly above the parametric curve, as opposed to what we saw in the first case study. This accounts for why we see the difference in p-values. This pattern is consistent for smaller data sets.

To summarize, we found in all three studies that the reported results were significant at least at $p < 0.1$. In doing so, we found that using interpolation to estimate the p-values is generally a very good approximation to the parametric values and should therefore be used due to its computational efficiency. For generating the bounds, this was not always the case where the interpolated bounds

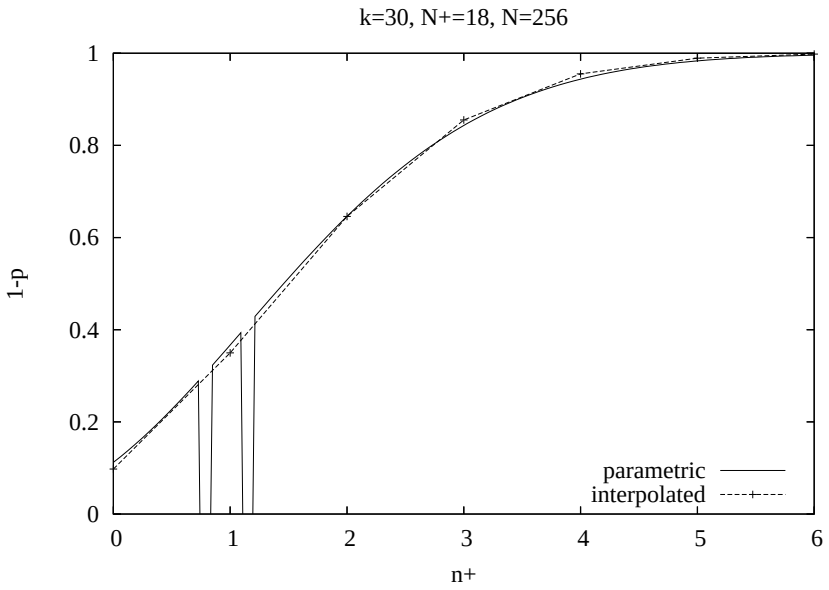

Figure 3: When the parametric function fails.

were shown to be slightly larger than the parametric bounds, whereas the discrete bounds were the most restrictive of them all. We further found that the parametric bounds were questionable for smaller data set sizes and should generally not be used unless you have large data sets and need the extra precision. Qualitatively, the difference between the discrete and parametric bounds were generally < 0.5, with the interpolated bounds generally being in between.

3.3 Finding Crossover Points

In this section we take advantage of the fact that we have access to the complete set of scores for the evaluation data set. We use these to perform a more global analysis over the complete set of scores. Specifically, we can investigate for which values of k the methods started to become significant for any given p. The top k bounds can be depicted in any kind of evaluation curve which is generated by varying the threshold of the model (*e.g.*, precision-recall curves, DET curves, lift-curves, etc.) ROC graphs plot false-positive (FP) rates (1–specificity) on the x-axis and true-positive (TP) rates (recall) on the y-axis. ROC curves are generated in a fashion similar to precision/recall curves, by varying a threshold across the output range of a scoring model, and observing the corresponding classification performances.[7] For this study, we use the ROC curves.

Figure 4 shows the ROC curves for the three learning methods as reported in the original study. The diagonal line, untitled, is the expected performance based on the random model at the $p = 0.5$ confidence level. The figure shows that all methods are clearly performing better than the random model.

As all methods showed decreasing values for p as k increased, we wanted to investigate their performances in terms of how they compare against the confidence bound at $p = 0.001$ for larger values of k. Figure 5 shows the ROC curves from the original study, focusing on the target area

7. For a good introduction on the use of ROC curves in Machine Learning, see (Fawcett, 2003).

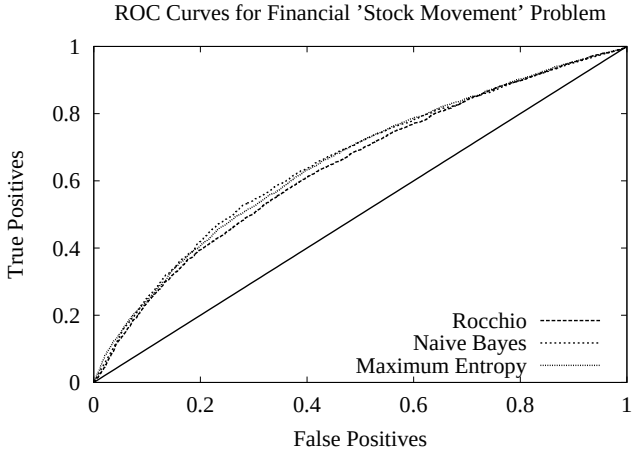

Figure 4: ROC curves for financial 'Stock Movement' problem

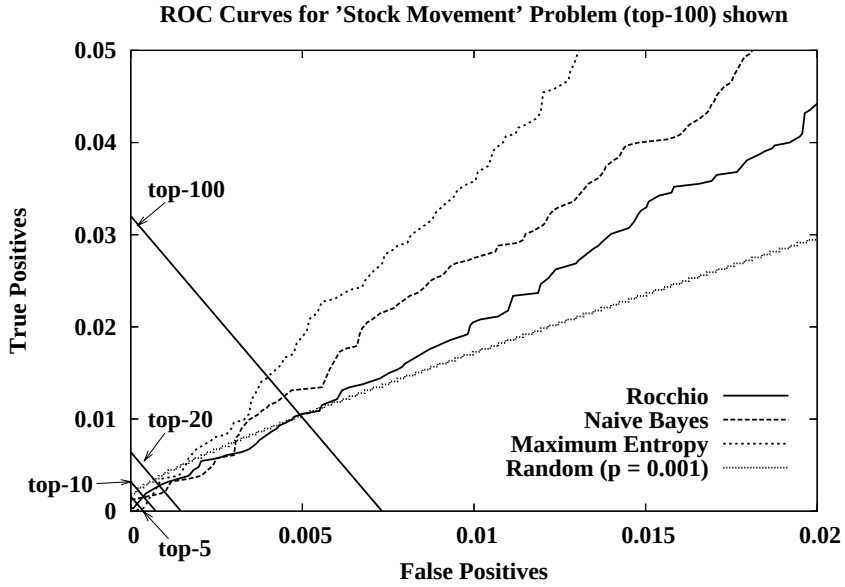

Figure 5: ROC curves for financial 'Stock Movement' problem showing top $\{100, 20, 10, 5\}$ lines.

where $k < 100$. It depicts, for $k = \{5, 10, 20, 100\}$, where the curves intersect the top k predictions. The figure shows 4 isolines, one for each value of k, where the isoline shows where all points for a given k would lie in ROC space.[8] The figure clearly shows that the methods quickly outpace the bound at $p = 0.001$ and are all quickly above that curve by a large margin.

We therefore generated the most significant bound we could pragmatically compute on our computers ($p = 10^{-17}$) without using slow arbitrary precision math libraries. The question we now investigate is whether any of the methods will ever become significant at such an extreme level.

8. Detailed explanations of these, and other, ROC isometrics for machine learning metrics can be found elsewhere (Provost & Fawcett, 2001; Flach, 2003).

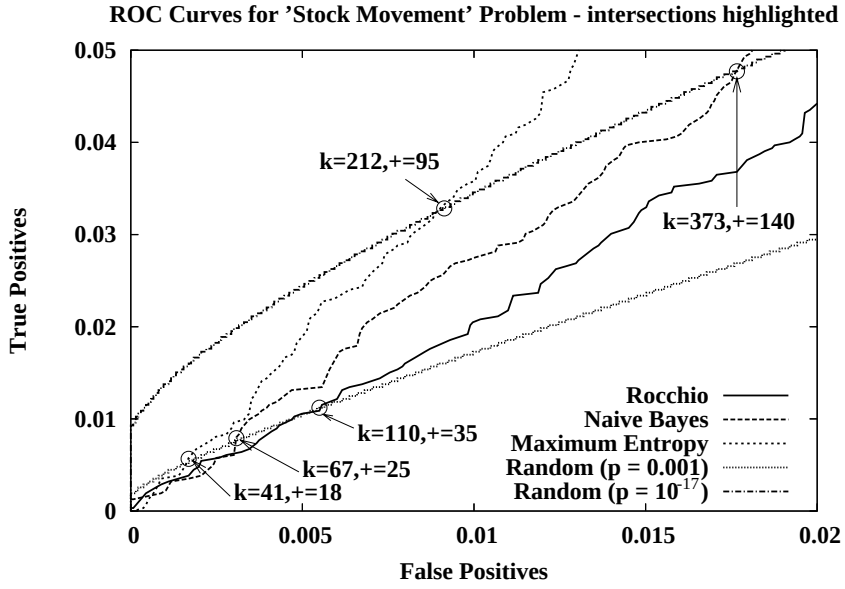

Figure 6: ROC curves for financial 'Stock Movement' problem showing where methods cross over to outperform $n(k,p)$ for $p = 0.001$ and $p = 10^{-17}$.

One question that can be answered by the technique used here is at which k does the performance of a method become significant. Let m_k be the number of positives method m has in its top k predictions. Given p and a model m, find k such that $m_k \leq n(k,p)$:

$$\text{cdf}(m_k, k) < (1-p) \leq \text{cdf}(m_k + 1, k) \tag{19}$$

It may be that the method crosses $n(k,p)$ more than once or follows $n(k,p)$ very closely, as is the case with Rocchio around $k = 100$ (see Figure 5). In such a case, it might make sense to strengthen the crossover criteria and specify that the method must perform as well as $n(k,p)$ over two or more contiguous k's. In this demonstration, we specify that the method must match $n(k,p)$ for at least 2 contiguous k's. Figure 6 highlights, given $p = 0.001$ and $[= 10^{-17}$, where the methods started outperforming each of the two $n(k,p)$ confidence bounds. Not shown in the graph is Rocchio's crossover of $n(k,p)$ at $p = 10^{-17}$. This happens at $k = 486$ with $m_k = 170$.

Finally, we can look at the bounds for all k and plot them in the same space as the evaluation curve—ROC space in this case. Performing such a visual comparison can give a more global and qualitative understanding of the performance of a system as well as quickly show where the systems perform significantly better than what would otherwise be expected. Figure 7 plots the same curves as before and adds the two random confidence bands, each generated by joining the bounds across all $k = 1, \ldots, N$ into one curve. While it was clear from the previous analysis that the models were *not* significant for $k < 20$, this figure clearly shows that all three learned models were well above the random confidence bounds for most k's, even at the extreme level of $p = 10^{-17}$.

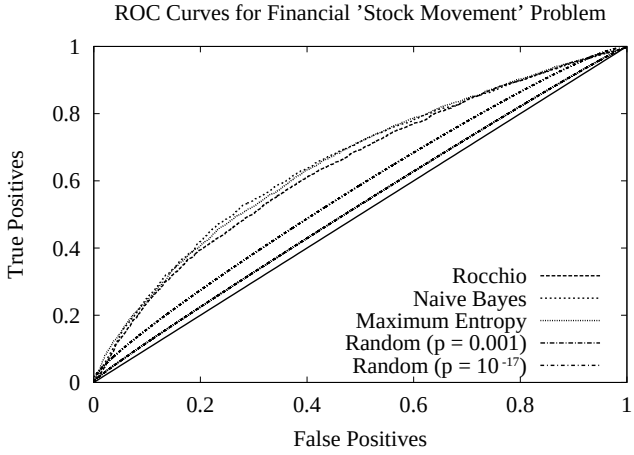

Figure 7: ROC curves for financial 'Stock Movement' problem with random confidence bands

4. Final Remarks

This paper introduced a new technique to efficiently generate confidence bounds for evaluation at top k predictions. It generates the bounds for a given k and p-value by finding the number of positives needed to dominate $(1-p)$ of all possible rankings for a given evaluation data set, $\mathcal{D}$.

The technique was demonstrated on three bodies of prior work, in which it was clearly shown that although the performances were generally greatly improved over the random model at $p = 0.5$, these improvements were not always as significant as might be implied by the increase in raw performance. Three important insights came out of this study: (1) it is hard to beat the random model at small values of k unless there is a large class skew, (2) interpolation of the p-values from the discrete bounds generally results in the same p-values as the more computationally intensive method of using parametric estimations, and (3) the parametric bounds fall apart for small data set sizes although they have slightly higher precision for larger data set sizes. Therefore, use of the interpolated bounds should be used unless higher precision is needed, in which case the parametric bounds should be used.

With regard to outperforming the random model at small values of k, we saw the obvious—that the more even the distribution, the more likely it is to see positives in the top k predictions. This has the effect seen in the analysis of the Hypertext'01 results where seeing 6 positives in the top 10 was only significant at $p = 0.100$. Such a situation makes it extremely difficult to outperform the random model at any significance level even though the qualitative improvements might seem impressive. This extreme density of positives lessens as the class skew becomes larger as is seen in the AAAI-1998 dataset where even seeing 2 positives in the top 10 is significant at $p = 0.026$.

Next, the technique was used to identify crossover points—the point where a method starts becoming significant for a given p. This was shown on the one data set where the full set of prediction scores were available. With this technique, it was possible to show that all three methods under consideration were able to outperform the random model at $p = 0.001$ relatively quickly and even for $p = 10^{-17}$ as k increased. This was shown even more strikingly when plotting the complete set of confidence bounds across all values of k.

The use of techniques such as the one described in this paper is important to verify that systems perform better than what would be expected from the random model. The demonstration clearly showed that even though the performances at first glance looked impressive, the differences were not necessarily as significant as the qualitative performance indicated.

Finally, the source code to compute the discrete bounds is available on the author's web-site. It is open source and requires Java 1.5.

Acknowledgments

Foster Provost helped improve this paper significantly through his many reviews, helpful suggestions and pointed questions. Michael Littman and Haym Hirsh both gave valuable feedback early in the design stages.

This work is sponsored in part by the National Science Foundation under award number IIS-0329135. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Science Foundation (NSF), or the U.S. Government.

References

Basu, C., Hirsh, H., Cohen, W. W., & Nevill-Manning, C. (2001). Technical Paper Recommendation: A Study in Combining Multiple Information Sources. *Journal of Artificial Intelligence Research*, *14*, 231–252.

Cohen, W. W. (1998). The WHIRL approach to information integration. In Hearst, M. (Ed.), *Trends and Controversies*. IEEE Intelligent Systems.

Cohen, W. W., Schapire, R. E., & Singer, Y. (1999). Learning to Order Things. *Journal of Artificial Intelligence Research*, *10*, 243–270.

Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the Society for Information Science*, *41*(6), 391–407.

Domingos, P., & Pazzani, M. (1996). Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 105–112.

Dumais, S., & Nielsen, J. (1992). Automating the assignment of submitted manuscripts to reviewers. In *Proceedings of the 15th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*.

Fawcett, T. (2003). ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. Technical report HPL-2003-4, HP Labs.

Flach, P. A. (2003). The geometry of ROC space: understanding machine learning metrics through ROC isometrics.. In *International Conference on Machine Learning (ICML'03)*.

Joachims, T. (1997). A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In Fisher, D. H. (Ed.), *Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 143–151, Nashville, US. Morgan Kaufmann.

Macskassy, S. A., Hirsh, H., Provost, F. J., Sankaranarayanan, R., & Dhar, V. (2001). Intelligent information triage. In *The 24th Annual Internationl Conference on Research and Development in Information Retrieval (SIGIR)*, New Orleans, LA.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*.

McCallum, A. K. (1996). Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/~mccallum/bow`.

Mitchell, T. (1997). *Machine Learning*. McGraw Hill.

Mitra, M., Singhal, A., & Buckley, C. (1998). Improving automatic query expansions. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia. ACM Press.

Nigam, K., Lafferty, J., & McCallum, A. (1999). Using maximum entropy for text classification. In *Proceedings of Machine Learning for Information Filtering Workshop, IJCAI'99*, Stockholm, Sweden.

Provost, F., & Fawcett, T. (2001). Robust Classification for Imprecise Environments. *Machine Learning*, *42*(3), 203–231.

Schapire, R., Singer, Y., & Singhal, A. (1998). Boosting and Rocchio applied to text filtering. In *Proceedings of ACM SIGIR*, pp. 215–223.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*(1).

Singhal, A., Abney, S., Bacchiani, M., Collins, M., Hindle, D., & Pereira, F. (1999). AT&T at TREC-8. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*.

Weisstein, E. W. (2002). Hypergeometric Distribution.. http://mathworld.wolfram.com/HypergeometricDistribution.html.