Identifying Patient Subgroups with Simple Bayes'

John M. Aronis[1], Ph.D., Gregory F. Cooper[2], M.D., Ph.D.,
Mehmet Kayaalp[2], M.D., M.S., Bruce G. Buchanan[1], Ph.D.,
[1]Department of Computer Science, University of Pittsburgh
[2]Center for Biomedical Informatics, University of Pittsburgh

Medical records can form the basis of retrospective studies, be used to evaluate hospital practices and guidelines, and provide examples for teaching medicine. Each of these tasks presumes the ability to accurately identify patient subgroups. We describe a method for selecting patient subgroups based on the text of their medical records and demonstrate its effectiveness. We also describe a modification of the basic system that does not assume the existence of a preclassified training set, and illustrate its effectiveness in one retrieval task.

INTRODUCTION

Hospital information systems can be invaluable to research and education [1]. For instance, retrospective studies can be based on medical records of patient subgroups, hospital practices and guidelines can be evaluated by examining the medical records of the affected patients, and historic data can provide important examples for teaching medicine. The ability to accurately and easily identify patient subgroups is essential to each of these applications. Many patient subgroups can be identified by a simple boolean query of coded data fields. However, currently many important patient subgroups can only be identified using the text portions of their medical records.

Simple Bayes' systems developed within the Machine Learning community have been successfully used to classify text documents [2]. In this paper we describe a Simple Bayes' system, report the results of experiments that support its effectiveness on medical records, and illustrate its application to the problem of identifying patient subgroups from text medical records. This work was performed in the context of the MARS (Medical ARchival System) at the University of Pittsburgh Medical Center [3]. MARS has been the primary tool for identifying patient subgroups in the UPMC research community; the present work aims to increase its functionality, convenience, and usage.

TEXT SIMPLE BAYES'

In this section we describe our system, Text Simple Bayes' (TSB). TSB constructs a model from a set of preclassified documents, then applies this model to a set of previously unseen documents. TSB can be used in either of two ways: it can predict the most likely class of each document in the test set, or it can rank the documents in the test set according to their estimated probability of being in a particular class.

Bayes' Rule underlies most modern AI systems that compute probabilistic inference. Using the notation of our domain, where $P(C|D)$ is the probability that C is the class of document D, $P(D|C)$ is the probability that an item in class C is expressed as document D, $P(C)$ is the prior probability that a document belongs to class C, and $P(D)$ is the probability that a randomly selected document will be of interest to the user, we can state Bayes' Rule as:

$$P(C|D) = \frac{P(D|C)P(C)}{P(D)} \qquad (1)$$

We make two assumptions that will allow us to convert Equation 1 into a useful form. First, we assume that a document can be accurately indexed and retrieved based solely on the unordered set of words that occur in it. And second, we assume that words occur in a document independently, conditioned on the class of the document. Of course, these assumptions are unlikely to hold exactly. The first assumption essentially says that word order and usage do not matter. The second assumption says that patterns of co-occurrence do not matter. Nonetheless, Simple Bayes' models have been shown to perform well [1,4].

Suppose we have a document D that contains words drawn from the vocabulary $w_1, \ldots, w_n$. For each w_i we construct a binary feature f_i that indicates whether or not w_i appears in the document.

1091—8280/99/$5.00 © 1999 AMIA, Inc.

Under the assumptions above, we can derive a formula for the *maximum a posteriori* (MAP) hypothesis:

$$C_{MAP} = \arg\max_{C} P(C) \prod_{i} P(f_i|C) \qquad (2)$$

Thus, C_{MAP} is the most probable class of the document D as computed by Bayes' Rule under the assumptions just stated. Using these assumptions, we can also derive the probability that a document belongs to a certain class:

$$P(C|D) = \frac{P(C)\prod_i P(f_i|C)}{\sum_j (P(C_j)\prod_i P(f_i|C_j))} \qquad (3)$$

where C is the class of interest (as specified by the user) and C_j ranges over the set of classes.

Both Equation 2 and Equation 3 are useful because the probabilities on the right are easy to estimate from historical data and can be combined to form a reliable estimate of the quantity on the left. Specifically, given a set of previously classified documents, we estimate $P(f_i|C)$ by:

$$P(f_i|C) \approx \frac{frequency(f_i, C) + 1}{frequency(C) + 2} \qquad (4)$$

where $frequency(f_i, C)$ is the number of times that f_i and C co-occur in the training data and $frequency(C)$ is the number of times that C occurs in the training data. Notice that $P(f_i|C)$ approaches 1/2 for infrequent words and small classes. We use this method because it smooths the estimates for words that are rare and/or training sets that are small; other smoothing operations are certainly possible. We also estimate $P(C)$ by:

$$P(C) \approx \frac{frequency(C) + 1}{N + m} \qquad (5)$$

where N is the total number of documents in the training data and m is the number of states that C can have. In general, these probability estimates are simple, efficient to compute, and robust.

We use a simple form of feature selection. The *likelihood ratio* of a word w for a class C is:

$$lr(w) = \frac{P(w|C)}{P(w|not\text{-}C)}$$

We define the ***adjusted likelihood ratio*** by $alr(w) = lr(w)$ if $lr(w) \geq 1$, or $alr(w) = 1/lr(w)$ otherwise. Therefore, $alr(w)$ is large if w correlates with the class or its complement. Starting with W, the set of words that occur in the entire training set, we construct the set of words $W_{m,n}$ by 1) eliminating words from W that occur in fewer than m documents, 2) ordering the remaining words according to alr, and 3) taking the n words with the highest alr value. We can select values for m and n by subdividing the training set into a smaller train/test partition and selecting m and n based on their performance on this split, or we can select m and n based on an entirely separate dataset, or we can simply guess values based on our knowledge of the domain.

THE EFFECTIVENESS OF TSB

In this section we describe experiments that investigate the effectiveness of TSB on the text of medical records. We have conducted these experiments using standard machine learning methodologies for two reasons. First, results established in this manner usually generalize to other applications of the basic method. And second, we wish to facilitate comparisons with other systems and domains in the machine learning literature. In the next section we describe a modification of the basic TSB system that performs well in more realistic situations.

The first dataset consists of discharge summaries of 2,060 patients admitted to two medical ICU's at the University of Pittsburgh Medical Center, between January 1, 1993 and December 31, 1995. Of these patients, 80 were identified as having venous thrombosis (VT) based on the following ICD-9 codes at discharge: other venous thrombosis (453), Budd-Chiari syndrome (453.0), thrombophlebitis migrans (453.1), vena cava syndrome (453.2), renal vein thrombosis (453.3), venous thrombosis nec (453.8), and venous thrombosis nos (453.9). (The vast majority of these cases were deep venous thrombosis.) These records were reviewed by an ICU specialist who verified that the records support a VT diagnosis. Thus, the dataset consists of 80 records in class *VT* and 1,980 records in class *not-VT*. All of the records are text. The typical record contains about 1,500 words and the entire dataset contains about 13,000 unique words. We processed the text to remove numbers and punctuation. As an additional layer of security to protect confidentiality, we removed possible patient identifiers from the text. This was done by first removing sequences of capitalized words that begin with a title (such as *Mr.*, *Ms.*, *Mrs.*, or *Dr.*), then removing capitalized words that do not appear in the *Unified Medical Language System*. This method is effective, although certainly not perfect and does not replace other protections.

Our experiments with this dataset consisted of the following steps:

1. We randomly divided the records into two sets, putting 60% of the records in the *Train* set, and 40% of the records in the *Test* set. We balanced the fraction of *VT* and *not-VT* records in these sets. Thus, *Train* consisted of 48 *VT* records and 1188 *not-VT* records, and *Test* consisted of 32 *VT* records and 792 *not-VT* records.

2. We further divided *Train* into *TrainTrain* and *TrainTest*, putting 60% of *Train* in *TrainTrain* and the remaining 40% in *TrainTest*, again balancing the fraction of *VT* and *not-VT* records in each set.

3. We used the *TrainTrain/TrainTest* split for feature selection. Specifically, we searched for values of m and n such that when TSB was trained on *TrainTrain* using only the words in $W_{m,n}$ it performed best when tested on *TrainTest*.

4. We selected the set of words $W_{m,n}$ from all of *Train* using the values of m and n determined in the previous step, then calculated the probabilities in Equation 4 and Equation 5 from *Train* using these words.

5. We used the probabilities from Step 4 to make predictions for the records in *Test*.

The contingency table in Table 1 illustrates the result of using the MAP hypotheses computed by Equation 2 to predict the class of each record in *Test*. Simple accuracy is easily computed from this table to be 94%. However, this apparently high number is misleading since the accuracy of always making the default assumption (*not-VT*) for every record in *Test* is 96%. We believe the MAP hypothesis yielded an accuracy no higher than 94% at least in part due to the small number of *VT* cases in the training set. Importantly, however, we do not expect the machine to autonomously select medical records. In particular, we want the system to rank records for review by the user such that probable *VT* records are presented before probable *not-VT* records.

	True Class	
Predicted Class	*VT*	*not-VT*
VT	21	37
not-VT	11	755

Table 1: Contingency table for *VT*.

We can use Equation 3 to rank records in *Test* according to the probablity that they are in *VT*, placing highly probable *VT* records near the beginning of the ordering. When the user reviews the records in order, they hopefully will encounter a large number of ***hits*** (records of the type they are seeking—here *VT*) early in the process, and can simply stop when the frequency of hits gets too low. Each *VT* case reviewed is a ***true-positive***, and each *not-VT* case reviewed is a ***false-positive***. A *Receiver Operating Characteristic* (ROC) curve plots the true-positive rate (on the y-axis) against the false-positive rate (on the x axis); we can measure the efficiency of a ranking by the area under the ROC curve [5]. The area under the ROC curve generated by Equation 3 on this dataset is 0.93. (The area under the ROC curve generated by a random ordering is about 0.50.)

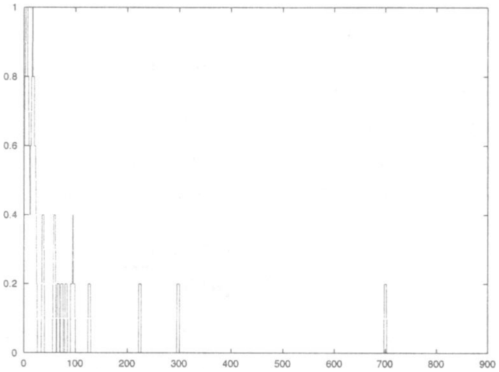

Figure 1: Density of VT hits.

We can also visualize the ordering with the ***density graph*** in Figure 1. This plots the ordering on the x-axis (with the first record in the ordering on the left) and ***density*** of hits (the fraction of hits in a window of width 5) on the y-axis. As you can see, the ordering contains a high density of hits towards the beginning, and only a few stragglers near the end. By reviewing cases in this order a user can find *VT* cases very efficiently, and stop when the density decreases below a level that justifies further search.

The second dataset we used consists of discharge summaries of 168 patients who were seen at the University of Pittsburgh Medical Center or a local affiliate between January 1, 1995 and January 31, 1999 who had the term *alteplase*[1] in their pharmacy discharge summary. (The pharmacy discharge summary is different than the general hospital discharge summary.) Of these patients, 94 were identified as experiencing either ***acute, but ill-defined, cerebrovascular disease***, or ***acute my-***

[1] Alteplase is indicated for use in the management of acute MI and acute ischemic stroke in adults.

ocardial infarction, based on ICD-9 codes in the 436 or 410 range or DRG codes 121, 122, 123, or 014. (The medical records of this combined set of patients were requested by a MARS user for their research.) Thus, the dataset consists of 94 records in class *CVA-MI* and 74 records in class *not-CVA-MI*. All of the records were text. The typical record contained about 750 words and the entire dataset contained about 5,000 unique words. We processed the text to remove numbers, punctuation, and proper names.

	True Class	
Predicted Class	*CVA-MI*	*not-CVA-MI*
CVA-MI	35	9
not-CVA-MI	3	21

Table 2: Contingency table for *CVA-MI*.

We derived the previous performance metrics using this dataset. Here, the 168 records were divided into a *Train* set consising of 100 records (56 *CVA-MI* and 44 *not-CVA-MI*), and a *Test* set consisting of 68 records (38 *CVA-MI* and 30 *not-CVA-MI*). *Train* was further divided into *Train-Train* and *TrainTest* for feature selection. The resulting contingency table is shown in Table 2. This has accuracy of 82%, which is considerably better than always making the default conclusion of *CVA-MI* with accuracy 56%. The ROC curve has area 0.88, and the density graph is shown in Figure 2.

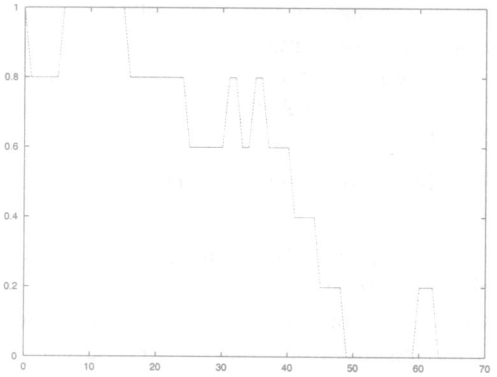

Figure 2: Density of CVA-MI hits.

COMPUTER-ASSISTED SEARCH FOR PATIENT SUBGROUPS

The experiments in the previous section were intended to investigate whether Simple Bayes' modeling is effective with the text of medical records. There are a variety of ways in which the basic TSB system described can be modified to fit the needs of actual applications. We present one such modification based on the work reported in [1]. This approach uses a two-stage method to identify the medical records of a patient subgroup. First, an initial computer model is constructed to rank patient records. Second, an iterative process is started in which the current model is used to rank patient records, the user examines and classifies the most likely hits according to this ranking, a new model is built using the newly classified records, and the process is repeated.

We do not always have a training set of preclassified records available to build an initial model. This is not a problem if the records the user is looking for are common—he or she can simply search through a sequence of records until enough records of interest are accumulated to build an initial model; thereafter, TSB can be used to build better models to locate records at an even faster rate. However, if the target class is relatively rare, this initial phase will be difficult and frustrating. For instance, *VT* is present in less than 5% of the records, so the user will need to examine at least 20 records for each hit.

In order to ***prime*** the process we can build an initial model based on keywords specified by the user. If the user specifies keywords $w_1, \ldots, w_n$ and class C, we can define an initial (artificial) training set that has one record in class C that consists of exactly the words $w_1, \ldots, w_n$, and one record in class *not-C* that is empty. Using this initial training set, we compute the following initial probabilities using Equation 4:

$$\begin{aligned} P(w_i \text{ occurs}|C) &= 2/3 \\ P(w_i \text{ does not occur}|C) &= 1/3 \\ P(w_i \text{ occurs}|\textit{not-C}) &= 1/3 \\ P(w_i \text{ does not occur}|\textit{not-C}) &= 2/3 \end{aligned}$$

When records are ranked according to the probabilities computed by Equation 3, these estimates will cause records that contain all or most of the keywords to be ranked before records that contain few or none of the keywords.

The following algorithm starts with a set of keywords supplied by the user and a set of unclassified *Test* records. We assume the user would be willing to examine on each iteration the 40 records that the system rates as mostly likely to be hits.

1. Start with *Train* equal to the null set. Build an initial model from keywords supplied by the user as described above.

2. Order the records in *Test* using the model.
3. The user reviews and classifies the first 40 records in the ordering, removes hits and places them in *Train*.
4. Build a new model with the new *Train* set.
5. Go to 2.

Thus, the user will classify 40 new records on each iteration. Ideally, the ordering in Step 2 would always start with 40 records from the target class allowing the user to score 40 hits on each cycle.

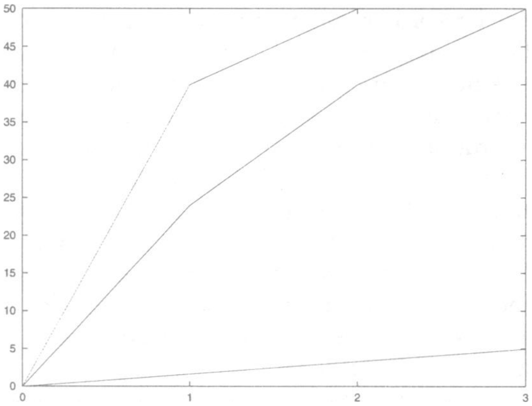

Figure 3: Number of VT cases located.

Figure 3 shows the performance of this algorithm on a test set consisting of 50 *VT* and 1,168 *not-VT* records (these numbers are based on the experiments reported in [1]). The records were preclassified as *VT* or *not-VT* and we simulated the actions of a user examining records on each iteration. The following keywords were used to construct the initial model: ***doppler, deep, venous, thrombosis, dvt, greenfield, filter, anticoagulation, heparin.*** The iteration number is plotted on the x-axis, and the cumulative number of *VT* cases located is plotted on the y-axis. The top line is the number of cases that would be located by an ideal system (i.e., one that located 40 *VT* cases on the first iteration, and the remaining 10 on the next iteration), the middle line is the number of cases located by TSB and the algorithm described above, and the bottom line is the number of cases that would be located by random selection.

DISCUSSION

We have presented data to support the claim that Simple Bayes' can form the basis of effective retrieval and classification systems for free-text medical records, despite the assumptions made by Simple Bayes' systems. Furthermore, Simple Bayes' systems avoid most of the problems associated with natural language processing by representing text with simple data structures. Finally, Simple Bayes' systems are efficient and robust, which is especially important for real-time or interactive applications that model with a large number of records.

However, while Simple Bayes' systems clearly are useful now, and will be for quite a while, we do not believe they represent the ultimate form of text processing for information retrieval. Simple Bayes' and other ***bag-of-words*** approaches to text processing cannot represent time, semantic relationships, negations, or modalities, and their range of applicability is limited. For the purpose of patient subgroup identification, it remains to be seen whether text-skimming methods will be adequate, or if methods more closely akin to full natural language understanding are necessary.

ACKNOWLEDGEMENTS

This work was supported by IAIMS grant 1-G08-LM06625 from the National Library of Medicine to the University of Pittsburgh. We thank Melissa Saul and Amanda Shultz of the University of Pittsburgh Medical Center Health System, and Dr. John K. Vries of the University of Pittsburgh School of Medicine.

REFERENCES

1. Cooper GF, Buchanan BG, Kayaalp M, Saul M, Vries JK. Using computer modeling to help identify patient subgroups in clinical data repositories. In: Proceedings of the American Medical Informatics Association Annual Symposium, 1998; pp. 180–184.

2. Mitchell TM. Machine learning. McGraw-Hill; 1997.

3. Yount RJ, Vries JK, Councill CD. The Medical Archival System: an information retrieval system based on distributed parallel processing. Information Processing Management, 1991; 27:379.

4. Domingos P, Pazzani M. On the optimality of the simple Bayesian classifier under zero-one loss. Machine Learning, 1997; 29:103–130.

5. Provost F, Fawcett T. Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions In: Proceedings of the International Conference on Knowledge Discovery and Data Mining, 1997; pp. 43–48.