Naive Bayes as a Satisficing Model

Ted Pedersen
Department of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75275–0122
`pedersen@seas.smu.edu`

Abstract

We report on an empirical study of supervised learning algorithms that induce models to resolve the meaning of ambiguous words in text. We find that the Naive Bayesian classifier is as accurate as several more sophisticated methods. This is a surprising result since Naive Bayes makes simplifying assumptions about disambiguation that are not realistic. However, our results correspond to a growing body of evidence that Naive Bayes acts as a satisficing model in a wide range of domains. We suggest that bias variance decompositions of classification error can be used to identify and develop satisficing models.

Introduction

The Naive Bayesian classifier (Duda & Hart 1973) makes very broad assumptions that usually do not correspond to a realistic model of the task at hand. Despite this, it proves remarkably successful in classification and prediction over a wide range of domains.

We present an empirical comparison of Naive Bayes and several other supervised learning algorithms that resolve the meaning of ambiguous words in text. Each method learns from training examples where the sense of an ambiguous word has been manually encoded. We find few significant differences between Naive Bayes and several more sophisticated methods that construct representative models of disambiguation.

This paper suggests that Naive Bayes is a satisficing model since it is an approximate representation of a domain and often proves as accurate as more complex and representative models that are constructed at much greater expense.

This paper begins with an introduction to Naive Bayes that includes a brief review of previous studies. We discuss word sense disambiguation and the role of Naive Bayes in past research. We outline our experimental design and present an extended discussion of our results disambiguating 12 words using 5 different algorithms. We close by pointing out that bias variance decompositions may offer a means of identifying and developing satisficing models.

Naive Bayes

The Naive Bayes model assumes that a set of features are all conditionally independent given the value of a classification variable. In disambiguation the contextual features of a sentence are represented by variables $(F_1, F_2, \ldots, F_n)$ and the sense of the ambiguous word is represented by (S). Contextual features in Naive Bayes only interact with the sense variable and do not interact directly with each other.

The probability of observing a particular sense in a given context is computed as follows:

$$p(S, F_1, F_2, \ldots, F_n) = p(S) \prod_{i=1}^{n} p(F_i|S) \qquad (1)$$

Training the Naive Bayes model with examples is a relatively simple process. The parameters $p(F_i|S)$ are estimated from a corpus of sense–tagged text. Even with a large number of features the number of parameters in Naive Bayes is comparatively small. For a problem with I features, each having L possible values, and a classification variable with K possible values, the number of parameters to estimate in the Naive Bayes model is $I * L * K$. There are I interactions among features.

Previous Work Naive Bayes has accumulated a considerable record of success in a range of domains. For example, (Clark & Niblett 1989) compare Naive Bayes, a rule induction system, and a decision tree learner. They find that Naive Bayes performs as accurately as these more sophisticated methods in various medical diagnosis problems.

A more extensive study of Naive Bayes appears in (Langley, Iba, & Thompson 1992). They compare Naive Bayes and a decision tree learner using data from the UCI Machine Learning repository (Merz, Murphy, & Aha 1997). For 4 of 5 naturally occurring data sets they report that Naive Bayes is the more accurate. They also present an average case analysis of Naive Bayes that is verified empirically using artificial data.

(Provan & Singh 1996) compare Naive Bayes and more complex Bayesian networks that diagnose the cause of acute abdominal pain. They argue that simple classification models will often outperform more detailed ones if the domain is complex and the amount of data available is relatively small. Their experiment consists of 1270 cases, each of which has 169 features. They find that the Naive Bayes model with 169 interactions is more accurate than a more detailed Bayesian network that has 590 interactions.

(Pazzani, Muramatsu, & Billsus 1996) discuss a software agent that learns to rate Web pages according to a user's level of interest. They construct a profile using examples of pages that a user likes and dislikes. They apply a number of learning algorithms and find that Naive Bayes is most accurate at predicting Web pages a user will find interesting.

Finally, (Domingos & Pazzani 1997) compare the accuracy of Naive Bayes with a decision tree learner, a nearest neighbor algorithm, and a rule induction system. They report that Naive Bayes is at least as accurate as the rule induction system and nearest neighbor algorithm for 22 of 28 UCI data sets and at least as accurate as the decision tree learner for 20 of 28 data sets. They also present an extensive analysis of the conditions under which Naive Bayes is an optimal classifier even when the conditional independence assumptions are not valid.

Word Sense Disambiguation

A fundamental problem in any text mining or natural language processing application is the ambiguity of word meanings. For example, *bill* has a variety of senses – a piece of currency, a proposed law, the jaws of a bird, etc. In *The Senate bill is being voted on*, it is clear to a human reader that *bill* is used in the legislative sense. However, a text mining agent searching for facts about bird anatomy might erroneously select this sentence unless *bill* is disambiguated. Automatically annotating text with sense distinctions of ambiguous words can improve document classification (e.g., (Voorhees 1993), (Schütze & Pedersen 1995)) and enhance the performance of Web–mining agents like those described in (Etzioni 1996).

Word sense disambiguation is frequently approached as a problem in supervised learning (e.g., (Gale, Church, & Yarowsky 1992), (Leacock, Towell, & Voorhees 1993), (Bruce & Wiebe 1994b), (Mooney 1996), (Ng & Lee 1996), (Ng 1997), (Pedersen & Bruce 1997), (Pedersen, Bruce, & Wiebe 1997)). A model of disambiguation is induced from a corpus of text where the sense of ambiguous words have been manually encoded. Most of these methods result in detailed and representative models of the context in which the ambiguous word occurs. This model is used to disambiguate instances of the ambiguous word that are subsequently encountered.

Under Naive Bayes, the model of disambiguation simply assumes that no two contextual features in a sentence will directly affect each other. This is not a realistic assumption for many linguistic features. For example, suppose we wish to predict the part of speech of a word at position i in a sentence, based on the part of speech at position $i-1$. Clearly there is a relationship between these two features. If position $i-1$ is an article, then it is more likely that position i is a noun or an adjective rather than another article since constructions such as ... *the a* ... are unusual at best.

Several other comparative studies of word sense disambiguation find Naive Bayes to be competitive with more sophisticated learning algorithms (e.g., (Leacock, Towell, & Voorhees 1993), (Mooney 1996), (Ng 1997)). These studies differ from ours in that they employ a feature set, commonly known as bag–of–words, that consists of thousands of binary features, each of which indicates the presence or absence of a particular word within some fixed distance of the ambiguous word.

Previous interpretations of the success of Naive Bayes focus on the bag–of–words (e.g., (Pedersen & Bruce 1997)). Given so many features, the assumptions of conditional independence made by Naive Bayes seem reasonable and may result in a model that represents the training data reasonably well. However, an earlier study of the feature set used in this paper shows that there are models other than Naive Bayes that better characterize our data (Pedersen, Bruce, & Wiebe 1997). Thus the assumptions of conditional independence made by Naive Bayes seem less appropriate here.

Experimental Design

This section provides an overview of the text being disambiguated, the features used to represent sentences with ambiguous words, and the algorithms that will be compared to Naive Bayes.

Text

The data used in these experiments is a sense–tagged corpus created by Bruce and Wiebe that is described in greater detail in (Bruce & Wiebe 1994a), (Bruce & Wiebe 1994b), (Bruce 1995), and (Bruce, Wiebe, & Pedersen 1996). They selected the following 12 words from the ACL/DCI Wall Street Journal corpus (Marcus, Santorini, & Marcinkiewicz 1993):

- Nouns: *interest*, *bill*, *concern*, and *drug*.
- Verbs: *close*, *help*, *agree*, and *include*.
- Adjectives: *chief*, *public*, *last*, and *common*.

They extracted every sentence containing one of these words and manually tagged the ambiguous word with a sense from the Longman Dictionary of Contemporary English (LDOCE) (Procter 1978). The number of sentences where each word occurs as well as the number of possible senses are shown in Figure 2.

Feature Set

Each sentence in the sense–tagged corpus is reduced to a feature vector (POS_{-2}, POS_{-1}, POS_{+1}, POS_{+2},

	C_1	C_2	C_3
agree	million	that	to
bill	auction	discount	treasury
chief	economist	executive	officer
close	at	cents	trading
common	million	sense	share
concern	about	million	that
drug	company	FDA	generic
help	him	not	then
include	are	be	in
interest	in	percent	rate
last	month	week	year
public	going	offering	school

Figure 1: Co–occurrence feature values

CO_1, CO_2, CO_3, MORPH, SENSE). This feature set was developed by Bruce and Wiebe in the work cited above. These features are defined as follows:

$\mathbf{POS}_X$ represents the part of speech of words that occur 1 or 2 positions to the left (-) or right (+) of the ambiguous word. The part of speech tags are derived from the first letter of the tags in the ACL/DCI WSJ corpus and have 25 possible values.

$\mathbf{CO}_X$ is a binary feature representing a co–occurrence. These features indicate whether or not a particular word occurs in the same sentence as the ambiguous word. They were selected from among the 400 words that occur most frequently in the sentences containing the ambiguous word. The three words chosen are the most indicative of the sense of the ambiguous word as judged by a test for independence. The values of this feature for each word are shown in Figure 1.

MORPH represents the morphology of the ambiguous word. It is a binary feature for nouns; representing if the word is singular or plural. It indicates the tense of the verbs and has between 2 and 7 possible values. This feature is not used for adjectives.

SENSE represents the sense of an ambiguous word. The words in this study have between 2 and 7 possible LDOCE senses.

Learning Algorithms

We compare the following supervised learning algorithms with Naive Bayes.

Majority Classifier: All instances of an ambiguous word are assigned the most frequent sense in the training sample. This establishes a lower bound of disambiguation performance.

PEBLS (Cost & Salzberg 1993): A k nearest–neighbor algorithm where classification is performed by assigning an ambiguous word to the majority class of the k–nearest training examples. In these experiments each ambiguous word is assigned the sense of the single most similar training example (i.e., $k = 1$).

C4.5 (Quinlan 1992): A decision tree algorithm in which classification rules are formulated by recursively partitioning the training sample. Each nested partition is based on the feature value that provides the greatest increase in the information gain ratio for the current partition.

CN2 (Clark & Niblett 1989): A rule induction algorithm that selects classification rules that cover the largest possible subsets of the training sample as measured by the Laplace error estimate.

Naive Mix (Pedersen & Bruce 1997): A probabilistic classifier based on the averaged joint distribution of a sequence of decomposable models[1]. These models are generated during a forward sequential search. The best fitting model at each level of complexity in the search is included in the sequence. Complexity is measured by the number of interactions in the model and fit is evaluated by Akaike's Information Criteria (Akaike 1974).

Experimental Results

There are three experiments presented here. First, we perform 10–fold cross validation using all available data for each word and determine the accuracy of each method. We find few significant differences among the methods. Second, we vary the amount of training data in order to see how much is required to reach certain levels of accuracy. We find that Naive Bayes is less accurate when there are fewer than 300 training examples. Third, we decompose classification error into bias and variance components.

Cross Validation

Ten–fold cross validation is an efficient means of training and evaluation. All of the sense–tagged examples for a word are randomly shuffled and divided into 10 equal folds. Nine folds are used as training examples for a supervised learning algorithm. The remaining fold serves as a held–out test set to evaluate the learned model. This is repeated 10 times so that each fold serves as the test set once.

The average accuracy and standard deviation of each method across the 10 folds is reported in Figure 2. The row *win–tie–loss* shows the number of words that Naive Bayes disambiguates significantly more–equally–less accurately than the competing algorithm. Significance is judged by a pairwise t–test ($p = .01$).

The accuracy of Naive Bayes is not significantly different than the other methods that learn more detailed models at greater cost. This can quickly be confirmed by noting the average accuracy across all 12 words as well as the number of times the accuracy of Naive Bayes ties the other methods.

---

[1] Decomposable models were first applied to word sense disambiguation in (Bruce & Wiebe 1994b)

word/ # senses	sample size	Majority classifier	Naive Bayes	PEBLS k=1	C4.5	CN2	Naive Mix
chief/2	1040	.862 .026	.943 .015	.945 .018	.947 .020	.945 .013	.951 .016
common/6	1110	.802 .029	.832 .034	.853 .019	.871 .030	.803 .029	.853 .024
last/3	3180	.933 .014	.919 .011	.947 .012	.945 .008	.935 .013	.940 .016
public/7	870	.560 .055	.593 .054	.536 .039	.598 .047	.579 .057	.615 .055
adjectives	1550	.789	.822	.820	.840	.816	.840
bill/3	1340	.681 .044	.865 .026	.855 .034	.878 .029	.873 .035	.897 .026
concern/4	1490	.639 .054	.859 .037	.840 .036	.852 .042	.859 .033	.846 .039
drug/2	1220	.575 .033	.807 .036	.778 .034	.798 .038	.777 .069	.815 .041
interest/6	2360	.529 .026	.763 .016	.768 .020	.793 .019	.729 .034	.800 .019
nouns	1603	.606	.824	.810	.830	.810	.840
agree/3	1350	.777 .032	.930 .026	.928 .030	.947 .031	.947 .031	.948 .017
close/6	1530	.680 .033	.817 .023	.843 .042	.853 .021	.834 .036	.831 .033
help/4	1390	.753 .032	.780 .033	.710 .047	.790 .039	.779 .045	.796 .038
include/2	1560	.912 .024	.944 .021	.939 .015	.954 .019	.951 .018	.956 .018
verbs	1458	.781	.868	.855	.886	.878	.883
overall	1537	.725	.838	.829	.852	.834	.854
win-tie-loss		8-4-0		1-10-1	0-9-3	1-11-0	0-9-3

Figure 2: Disambiguation Accuracy

Learning Rate

The learning rate shows how accuracy changes as more training examples are utilized. We expect that increasing the number of examples will increase classification accuracy. However, there is a point at which further examples do not improve accuracy. Since our 10–fold cross validation results for the full sample did not result in many significant differences we would like to determine which algorithm reaches the highest accuracy with the fewest number of training examples.

We begin with very small amounts of training data; first 10 and then 50 examples. Thereafter we increment 100 examples at a time until all available training data is used. We induce a model using each quantity of training data and evaluate that model using a held–out test set. For each quantity of training data we perform a variant of 10–fold cross validation and divide the sense–tagged data into 10 folds. One fold is held out for evaluation and training examples are randomly sampled from the remaining 9 folds. This process is repeated 10 times so that each fold serves as the held–out test set once.

Figure 3 shows the learning rate of the Naive Mix, C4.5, Naive Bayes, and the Majority Classifier as the number of training examples is increased. Each plot shows the average accuracy with each method of the 4 words that belong to the indicated part of speech.

Adjectives C4.5 and the Naive Mix achieve nearly the same level of accuracy learning from 10 training examples as they do 900 examples. Naive Bayes has a "slower" learning rate. Accuracy is low with a small number of examples but quickly improves with the addition of training data. Naive Bayes achieves approximately the same accuracy as C4.5 and the Naive Mix after roughly 300 training examples. However, none of the methods significantly exceeds the accuracy of the Majority Classifier.

Nouns C4.5 and the Naive Mix are nearly as accurate as the Majority Classifier after only learning from 10 training examples. However, unlike the adjectives, accuracy increases with additional training data and significantly exceeds the Majority Classifier. Like the adjectives, Naive Bayes begins at very low accuracy but reaches the same accuracy as C4.5 and the Naive Mix after approximately 300 training examples.

Verbs As is the case with adjectives and verbs, Naive Bayes begins at a very low level of accuracy while C4.5 and the Naive Mix nearly match the accuracy of the Majority Classifier after only 10 training examples. All three methods exceed the Majority Classifier and perform at nearly exactly the same level of accuracy after approximately 600 examples.

The main difference among the methods in this experiment is that Naive Bayes has a slower learning rate; C4.5 and the Naive Mix achieve accuracy of the Majority Classifier usually with just 10 or 50 examples. This is at least partially due to the skewed sense distributions, especially for adjectives and verbs. Given these distributions of senses, 10 or 50 examples will contain enough information for C4.5 or the Naive Mix to learn the majority class and obtain that level of accuracy immediately. Naive Bayes is unable to do the same since it relies upon an assumed model.

Bias Variance Decomposition

Here we decompose classification error into two fundamental components. This allows us to make distinctions between approaches that perform at the same level of accuracy.

word	test	Naive Bayes			MC4		
	size	bias	var	error	bias	var	error
agree	750	.060	.013	.073	.044	.019	.063
bill	740	.134	.034	.168	.120	.035	.155
chief	440	.056	.007	.063	.033	.026	.059
common	510	.130	.032	.162	.109	.045	.154
interest	1760	.201	.070	.271	.178	.103	.281
close	930	.128	.045	.173	.122	.042	.164
last	2580	.058	.007	.065	.053	.006	.059
concern	890	.110	.040	.150	.131	.041	.172
drug	620	.178	.045	.223	.188	.014	.202
help	790	.170	.056	.226	.206	.034	.240
include	960	.053	.016	.069	.088	.006	.094
public	270	.338	.088	.426	.382	.033	.415

Figure 4: Bias Variance Estimates

Background Loss functions such as mean–squared error and misclassification error can be decomposed into two components, known as bias and variance. Bias is an estimate of the systematic error that a learning algorithm is expected to make. Variance estimates the degree to which the learning algorithm is affected by variations in the training sample.

(Geman, Bienenstock, & Doursat 1992) decompose mean–squared error into additive bias and variance components. This is a very useful tool in analyzing the results of regression problems. However, it not applicable to classification where error is measured by counting the number of misclassifications. Fortunately, numerous decompositions of misclassification error have recently been proposed (e.g., (Kong & Dietterich 1995), (Breiman 1996a), (Kohavi & Wolpert 1996), (Tibshirani 1996), (Friedman 1997), (James & Hastie 1997)).

Some learning algorithms are inherently unstable in that they produce very different models across various samples of training data that have only minor differences. These algorithms are said to have low bias and high variance; decision trees and neural networks are examples (Breiman 1996b).

High bias and low variance results in models that are more robust to changes in the training data but do not closely characterize the training data. Naive Bayes is generally regarded as high bias and low variance. This is because the assumptions it makes about the interactions among features have nothing to do with a particular training sample. Nearest neighbor algorithms such as PEBLS are also considered high bias and low variance. These methods classify not by inducing a model but rather by finding some number of closely matching examples in the training data.

Experiment We break down the classification error of Naive Bayes and a decision tree algorithm using the decomposition proposed in (Kohavi & Wolpert 1996). We choose this decomposition since it adheres to generalized loss–function decompositions developed in (Tibshirani 1996) and (James & Hastie 1997) and is also implemented in MLC++ (Kohavi, Sommerfield, & Dougherty 1996).

Generally, a bias estimate reflects how often the average classifications, across N training samples, of a learned model fails to correspond to the actual classifications in E. Variance estimates the degree to which the classification predicted by the learned model varies across multiple training samples. An algorithm that always makes the same classification regardless of the training data will have variance of 0. We follow the sampling procedure recommended in (Kohavi & Wolpert 1996) to estimate these values.

1. Randomly divide the data into two sets, D and E. Let the number of instances in $D = 2m$, where m is the desired size of the training sample. E serves as a held–out test set.
2. Generate N samples of size m from D.
3. Run each learning algorithm on the N training samples. Classify each observation in E using the learned model.
4. Repeat steps 1–3 R times to evaluate different E.

The learning algorithms in our experiment are Naive Bayes and MC4, the MLC++ version of C4.5. We choose MC4 since it is a decision tree learner and represents a low bias and high variance approach. By contrast, Naive Bayes is a high bias and low variance model.

We chose training samples of size $m = 300$ since this is the smallest number of examples with which all methods achieve comparable error rates. We set $D = 600$ and generate $N = 100$ training samples. We repeat this entire procedure $R = 10$ times.

We show the estimates of bias and variance obtained by MLC++ in Figure 4. The standard deviations across the 10 repetitions is not reported since it is always less than .001. This figure is divided into three

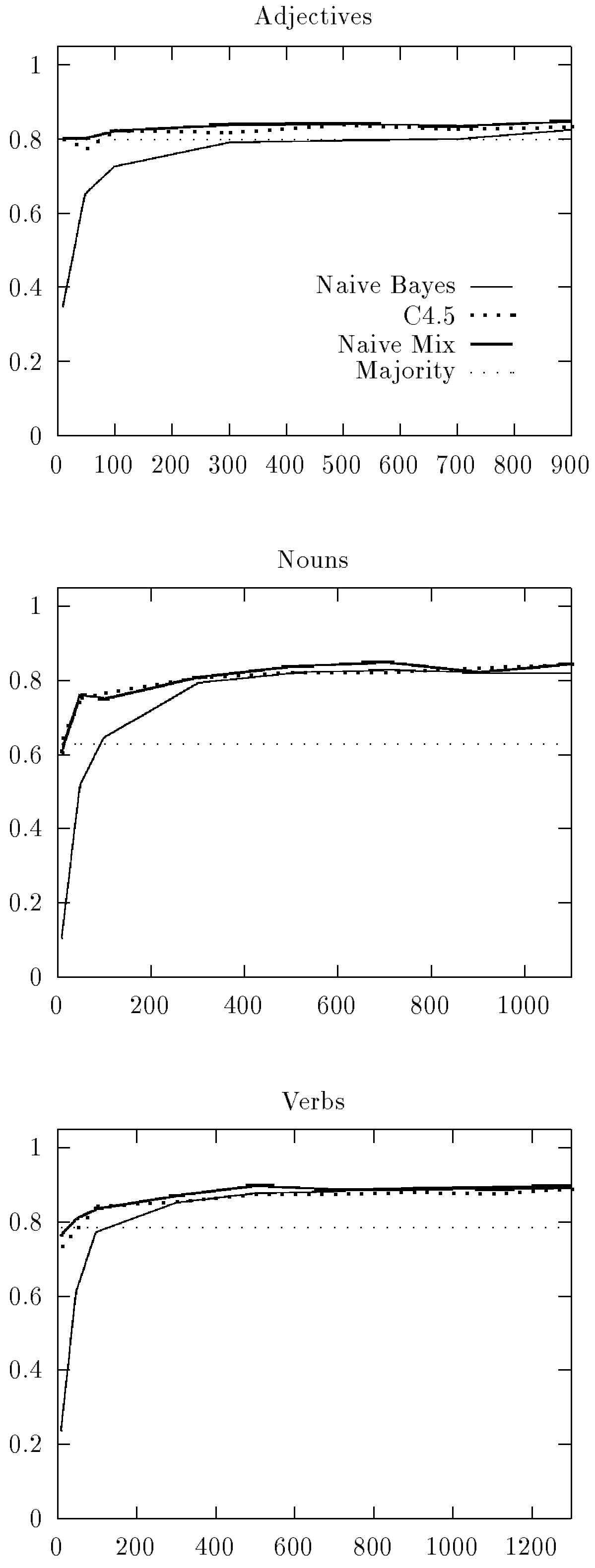

Figure 3: Training Examples versus Accuracy

groups of words. For the first group Naive Bayes has higher bias and lower variance than MC4. The second group of words has equal bias and variance for Naive Bayes and MC4. In the third group MC4 has higher bias and lower variance than Naive Bayes.

The data in Figure 4 is presented again as correlation plots in Figure 5. The bias, variance, and total classification error for a word is represented by a single point in each plot. Points on or near $x = y$ are associated with words who have nearly identical estimates for Naive Bayes and MC4.

The first plot shows the bias. There are no extreme differences between the bias of Naive Bayes and MC4. It may be that the training sample size of 300 is too small to bring out large differences in bias between the methods. We will examine the effect of varying the training sample size on bias and variance in future work. The second plot shows variance. We see more extreme differences, particularly for *public* and *interest*. The third plot in the figure shows the overall error rate, i.e., the sum of the bias and the variance. This confirms that the error rates for the two methods are fairly similar for each word.

Conclusions

We suggest the Naive Bayes model is a satisficing model for a complex and important problem in natural language processing. An empirical study shows that Naive Bayes is as accurate as methods that build more detailed models of disambiguation. The only exception to this behavior is when training sample sizes are very small. In those cases a decision tree learner and an averaged probabilistic model are able to take advantage of skewed sense distributions and duplicate the performance of the Majority Classifier.

Our analysis includes estimates of the bias and variance components of classification error. There is an inherent tradeoff between bias and variance. Bias is reduced by building more representative models of the training data. However, this leads to increases in variance since the more detailed models are sensitive to changes across different samples of training data.

Bias variance decompositions offer a systematic means of monitoring the tradeoff between representational power and robustness that are often made when building models. This is a potentially powerful tool for building and tuning satisficing models.

Acknowledgments

This research was supported by the Office of Naval Research under grant number N00014-95-1-0776.

References

Akaike, H. 1974. A new look at the statistical model identification. *IEEE Transactions on Automatic Control* AC-19(6):716–723.

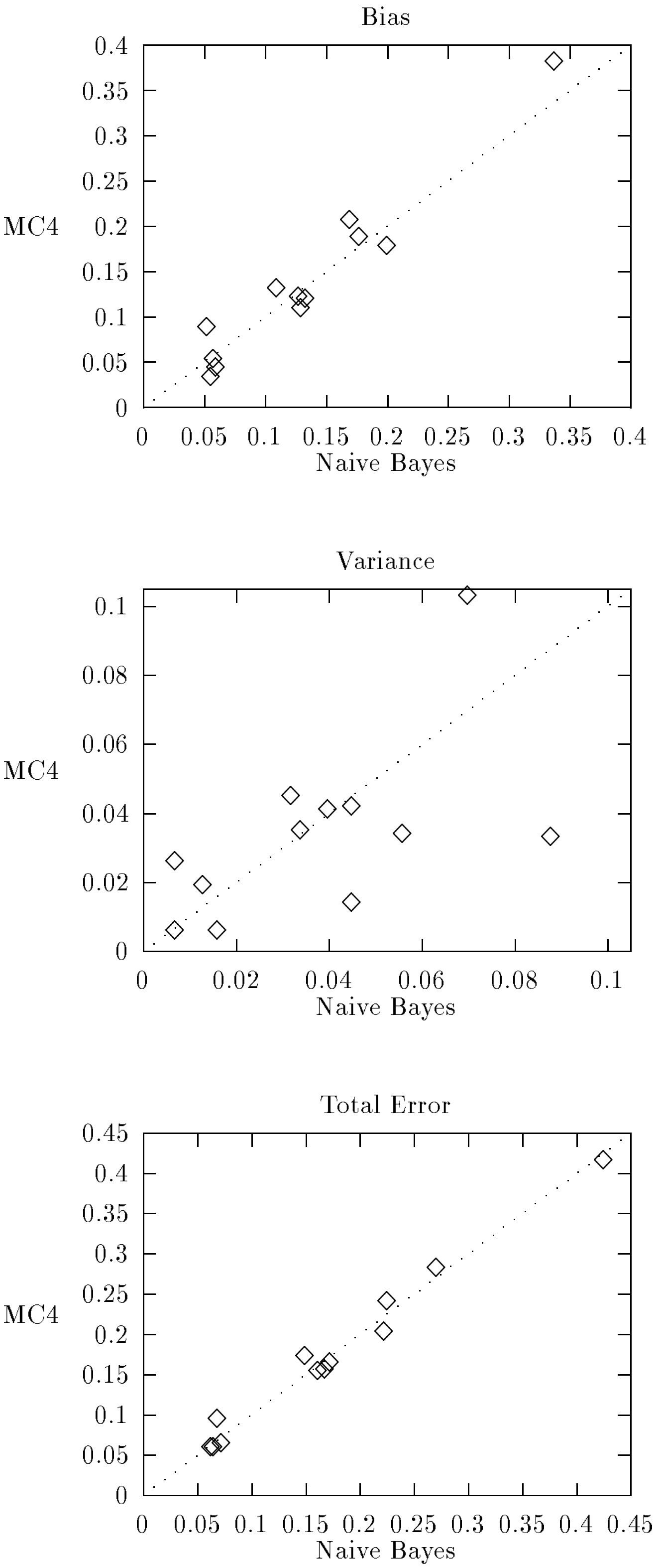

Figure 5: Bias, Variance, and Error Correlation

Breiman, L. 1996a. Bias, variance and arcing classifiers. Technical Report 460, University of Calfornia at Berkeley.

Breiman, L. 1996b. The heuristics of instability in model selection. *Annals of Statistics* 24:2350–2383.

Bruce, R., and Wiebe, J. 1994a. A new approach to word sense disambiguation. In *Proceedings of the ARPA Workshop on Human Language Technology*, 244–249.

Bruce, R., and Wiebe, J. 1994b. Word-sense disambiguation using decomposable models. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, 139–146.

Bruce, R.; Wiebe, J.; and Pedersen, T. 1996. The measure of a model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 101–112.

Bruce, R. 1995. *A Statistical Method for Word-Sense Disambiguation.* Ph.D. Dissertation, Dept. of Computer Science, New Mexico State University.

Clark, P., and Niblett, T. 1989. The CN2 induction algorithm. *Machine Learning* 3(4):261–283.

Cost, S., and Salzberg, S. 1993. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning* 10(1):57–78.

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

Duda, R., and Hart, P. 1973. *Pattern Classification and Scene Analysis.* New York, NY: Wiley.

Etzioni, O. 1996. The World–Wide Web: Quagmire or gold mine? *Communications of the ACM* 39(11):65–68.

Friedman, J. 1997. On bias, variance 0/1 loss, and the curse–of–dimensionality. *Data Mining and Knowledge Discovery* 1:55–77.

Gale, W.; Church, K.; and Yarowsky, D. 1992. A method for disambiguating word senses in a large corpus. *Computers and the Humanities* 26:415–439.

Geman, S.; Bienenstock, E.; and Doursat, R. 1992. Neural networks and the bias/variance dilemma. *Neural Computation* 4:1–58.

James, G., and Hastie, T. 1997. Generalizations of the bias/variance decomposition for prediction error. Technical report, Stanford University.

Kohavi, R., and Wolpert, D. 1996. Bias plus variance decomposition for zero–one loss functions. In *Proceedings of the Thirteenth International Conference on Machine Learning*, 275–283.

Kohavi, R.; Sommerfield, D.; and Dougherty, J. 1996. Data mining using MLC++: A machine learning library in C++. In *Tools with Artificial Intelligence*, 234–245. http://www.sgi.com/Technology/mlc.

Kong, E., and Dietterich, T. 1995. Error–correcting output coding corrects bias and variance. In *Proceedings of the 12th International Conference on Machine Learning*, 314–321.

Langley, P.; Iba, W.; and Thompson, K. 1992. An analysis of Bayesian classifiers. In *Proceedings of the 10th National Conference on Artificial Intelligence*, 223–228.

Leacock, C.; Towell, G.; and Voorhees, E. 1993. Corpus-based statistical sense resolution. In *Proceedings of the ARPA Workshop on Human Language Technology*, 260–265.

Marcus, M.; Santorini, B.; and Marcinkiewicz, M. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313–330.

Merz, C.; Murphy, P.; and Aha, D. 1997. UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/mlearn/MLRepository.html.

Mooney, R. 1996. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 82–91.

Ng, H., and Lee, H. 1996. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *Proceedings of the 34th Annual Meeting of the Society for Computational Linguistics*, 40–47.

Ng, H. 1997. Exemplar–based word sense disambiguation: Some recent improvements. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, 208–213.

Pazzani, M.; Muramatsu, J.; and Billsus, D. 1996. Syskill & Webert: Identifying interesting Web sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 54–61.

Pedersen, T., and Bruce, R. 1997. A new supervised learning algorithm for word sense disambiguation. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, 604–609.

Pedersen, T.; Bruce, R.; and Wiebe, J. 1997. Sequential model selection for word sense disambiguation. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, 388–395.

Procter, P., ed. 1978. *Longman Dictionary of Contemporary English.* Essex, UK: Longman Group Ltd.

Provan, G., and Singh, M. 1996. Data mining and model simplicity: A case study in diagnosis. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining.*

Quinlan, J. 1992. *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Schütze, H., and Pedersen, J. 1995. Information retrieval based on word senses. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, 161–175.

Tibshirani, R. 1996. Bias, variance and prediction error for classification rules. Technical report, University of Toronto.

Voorhees, E. 1993. Using WordNet to disambiguate word senses for text retrieval. In *Proceedings of SIGIR '93*, 171–180.