

Advances in Predictive Model Generation for Data Mining

Se June Hong and Sholom M. Weiss

IBM Research Report RC-21570

Advances in Predictive Model Generation for Data Mining

Se June Hong and Sholom M. Weiss

IBM T.J. Watson Research Center

P.O. Box 218, Yorktown Heights, NY 10598, USA

sjhong@us.ibm.com and sholom@us.ibm.com

Abstract

Predictive models have been widely used long before the development of the new field that we call data mining. Expanding application demand for data mining of ever increasing data warehouses, and the need for understandability of predictive models with increased accuracy of prediction, all have fueled recent advances in automated predictive methods. We first examine a few successful application areas and technical challenges they present. We discuss some theoretical developments in PAC learning and statistical learning theory leading to the emergence of support vector machines. We then examine some technical advances made in enhancing the performance of the models both in accuracy (boosting, bagging, stacking) and scalability of modeling through distributed model generation. Relatively new techniques for selecting good feature variables, feature discretization, generating probabilistic models, and the use of practical measures for performance will also be discussed.

Keywords: rules, decision trees, machine learning, error rate, loss function, boosting, feature selection, text mining, ROC curve, lift curve, probabilistic model, Bayesian network, neural network

1 Introduction

There are two types of models that are of interest. An explanatory model summarizes data for the purpose of making sense in terms of domain knowledge. A predictive model's main concern is the utility of the prediction when applied to the unseen, future cases. Traditionally, accuracy measures have been used almost exclusively in assessing the models for both classification and regression. Although there has been a long history of predictive model generation from statistics, more recent developments from machine learning area, including pattern recognition and artificial neural networks, have contributed to meeting the challenges of data mining ever increasing data warehouses. Predictive modeling, which is perhaps the most-used subfield of data mining, draws from statistics, machine learning, database techniques, optimization techniques, and theory of

learning. We want the models to be useful and accurate, practical to generate, and we often prefer models that are understandable.

The proven utility of industrial applications has led to recent advances in predictive modeling. We will discuss a few application areas that have demonstrated the importance of predictive modeling and have also fueled advances. Some important advances in learning theory will be considered in section 3. The idea of using multiple models to enhance the performance of a model has spawned several useful approaches with theoretical understanding for their success (section 4). One way to reduce the data and also reduce computational efforts of most model generation techniques, while often producing superior models, is to select a subset of relevant features from a much larger number of features, sometimes ranging into thousands. Section 5 discusses relatively recent development of wrapper techniques for feature selection and non-myopic feature merits that can be used for feature selection, and their use in non-myopic discretization of numeric features.

Scaling up to multi-giga bytes of data is one of the most important challenges of data mining. Besides the traditional sampling techniques, a new emphasis is being placed on fast disk-based algorithms, sometimes efficiently coupled into database management systems, as well as parallel and distributed algorithms. We will discuss these in section 6. Some new directions in the way of measuring the performance of a predictive model and of comparing different models will be discussed in section 7.

2 Challenging applications

It has been well-established that a substantial competitive advantage can be obtained by data mining in general and predictive modeling in particular. There are many success stories of predictive models mined from scientific data, financial data, and banking and insurance business data. Most of these reside in a large data warehouses. For some application area, maximizing accuracy or some other utility measure of the model is of paramount importance, even at the cost of giving up the understandability of a simple model. For some applications, the quality of available data in terms of missing values and noise present extra challenges for model generation. Most of the significant applications have large data volumes. We will briefly examine three challenging application areas: insurance, fraud detection, and text categorization.

2.1 Insurance

Risk assessment is in the core of the insurance business, where actuarial statistics have been the traditional tools of the trade in modeling various aspects of risk such as accident, health claims, or disaster rates, and the severity of these claims. The claim frequency is usually extremely rare and probabilistic by nature. For instance, the auto accident rate of an insured driver is never a clear no-accident class v.s. accident class problem and instead should be modeled as a Poisson distribution. The claim amounts usually follow a log-normal distribution which captures the phenomenon of rare but very high damage amounts. Neither of these distributions are well-modeled by conventional modeling tools such as CHAID [1], CART [2], C4.5 [3], SPRINT [4] or classical statistical regression techniques that optimize for traditional normal distributions. In general, different kinds of insurance are better modeled by using different statistical models depending on the fundamental nature of the claims process, requiring a predictive model that can be optimized for different underlying distributions.

Many other factors in insurance applications complicate the modeling process. Often, the desired target to be modeled is the expected claims amount for each individual policy holder, which is a joint distribution of the claim rate and claim amount that may range over several orders of magnitude among all claims. Stored records usually have a significant proportion of missing data or back filled data updated only at the time of accidents. The claims data usually has hundreds of fields, and demographic data must also be included. The total size of the policy holder database can be in the tens of Giga bytes. Insurance actuaries demand that the model must be actuarially credible, i.e. the parameters of the model be within 5% of the expected true value with 90% confidence.

The Underwriting Profitability Analysis (UPA) application [5] developed by Apte et al. embodies a new approach for generating predictive models for insurance risks addressing all the challenges above. It is based on their ProbeE kernel that directly solves for developing probabilistic models with any given underlying distribution. Risk groups are identified by a top down recursive splitting method similar to most tree generation algorithm. One of the key differences from the traditional tree splitting is that the splitting tests are selected by statistical models of insurance risk, e.g. joint Poisson and log-normal distribution for auto accident claim amount, otherwise known as pure premium. The splitting tries to optimize the maximum likelihood estimation (in this case, negative log likelihood) of all the examples given the assumed distribution. This application has yielded a demonstrably superior model in an actual large insurance firm data and many of the nugget rules, the leaf nodes of ProbeE trees, have replaced some of the existing actuarial rules. They also report that in this application, the model improves as more data are used in the training set, contrary to many applications which reach a plateau of performance after a few millions of examples.

2.2 Fraud detection

Fraud detection is an important problem because fraudulent insurance claims and credit card transactions alone cost tens of billions of dollars a year just in the US. In the case of credit card fraud, artificial neural-networks have been widely-used by many banks for some time. Attacking this problem is a major effort of many financial services institutions. Frauds are relatively rare, i.e. a skewed distribution that baffles many traditional data mining algorithms unless stratified samples are used in the training set. Some large banks add to the transaction data volume by millions of transactions per day. The cost of processing a fraud case, once detected, is a significant factor against false positive errors while undetected fraud adds to the transaction cost in the loss column. This not only calls for a more realistic performance measure than traditional accuracy, but also influences the decision whether to declare a transaction to be processed as a fraud or not. The pattern of fraudulent transactions varies with time (thieves are smart adapters) which requires relatively frequent and rapid generation of new models.

Recent progress in fraud detection has been reported for the JAL System (Java Agents for Meta-Learning) developed at Columbia University [6]. The massive set of data with binary labels of fraud or legitimate is divided into smaller subsets, for each participating bank units and for multiple samples to insure better performance. They produce models by some fast existing methods in a distributed fashion. These multiple base models are then combined to form a meta-learner [7] (see sections 4 and 6). Based on actual data from Chase bank and First Union bank in New York, the induced models produced cost savings about 25% over existing methods.

2.3 Text Mining

Electronic documents or text fields in databases are a large percentage of the data stored in centralized data warehouses. Text mining is the search for valuable patterns in stored text. When stored documents have correct labels, such as the topics of the documents, then that form of text mining is called text categorization.

In many carefully organized text storage and retrieval systems, documents are classified with one or more codes chosen from a classification system. For example, news services like Reuters carefully assign topics to news-stories. Similarly, a bank may route incoming e-mail to one of dozens of potential response sites.

Originally, human-engineered knowledge-based systems were developed to assign topics to newswires [8,9]. Such an approach to classification may have seemed reasonable ten years ago, but the cost

of the manual analysis needed to build a set of rules is no longer reasonable, given the overwhelming increase in the number of digital documents. Instead, automatic procedures are a realistic alternative, and researchers have proposed a plethora of techniques to solve this problem.

The use of a standardized collection of documents for analysis and testing, such as the Reuters collection of newswires for the year 1987, has allowed researchers to measure progress in this field. It can be empirically demonstrated that over the course of less than 10 years, substantial improvements in automated performance have been made.

Many automated prediction methods exist for extracting patterns from sample cases [10]. These patterns can be used to classify new cases. In text mining, specifically text categorization, the raw cases are individual documents. These cases can be transformed into a standard model of features and classes. The cases are encoded in terms of features in some numerical form, requiring a transformation from text to numbers. For each case, a uniform set of measurements on the features are taken by compiling a dictionary from the collection of training documents. The frequencies of occurrence of dictionary words in each document are measured. Prediction methods look at samples of documents with known topics, and attempt to find patterns for generalized rules that can be applied to new unclassified documents.

Given a dictionary, sample cases can be described in terms of the words or phrases from the dictionary found in the documents. Each case consists of the values of the features for a single article, where the values could either be boolean, indicating whether the feature appears in the text or does not, or numerical, some function of the frequency of occurrence in the text being processed. In addition, each case is labeled to indicate the classification of the article it represents. Once the data is in a standard numerical encoding for classification, any standard data mining method, such as decision trees or nearest neighbors, can be applied. Beside the issue of binary v.s. word count feature, another secondary characteristics of the dictionary words of interest is the issue of stemmed v.s. raw words, i.e., when compiling the dictionary, the words may be stemmed, i.e. mapped into a common root. For example, plural words may be mapped into their singular form.

A more critical characteristic of the dictionary is its size. How many words should the dictionary hold? The answer to this question depends on many factors including the prediction method and the available computing resources. A universal dictionary of all stemmed words in the document collection can be formed. A feature selection technique is sometimes used to select a small subset of words that are deemed relevant to a particular topic. For example, the Reuters collection has about 22,000 stemmed words. Feature selection for a given topic can yield a subset with less than 1000 words without any loss of performance.

One of the interesting challenges text mining poses is the problem of minimal labeling. Most text collections are not tagged with category labels. Human tagging is usually costly. Starting from some tagged examples, one wishes to develop a text categorization model by asking certain selected examples to be tagged, and one naturally wishes to minimize the number of such requests. Many approaches to this problem are being pursued by theorists as well as practical algorithm developers. Another interesting application of text mining technology is web-mining where a variety of features beyond the original text present a special challenge.

3 Theoretical advances

The theory of predictive modeling has evolved from two research communities: computational learning theory and statistical learning theory. Developments in these areas shed light on what kind of functions, i.e. mapping the feature vectors to the target values, can be learned efficiently with a given set of models. These results give an understanding of model complexity and how it can be used to assess the future performance of models on unseen data. These new concepts are beginning to guide the model search process as well as the model evaluation process for practical cases, complementing traditional techniques from statistics. Because predictive modeling generally assumes that unseen data will be independently and identically distributed (*iid*) as the training data, the theoretical analysis of the model is also confined to the same assumption. The ability to ascertain whether the model is truly applicable to new data seems to be beyond the scope of current research efforts.

Section 3.1 below is a brief summary of the description of these theoretical developments originally written from the statistical perspective in [11] by Hosking, Pednault and Sudan. The Support Vector Machine, discussed in section 3.2, is gaining increasing attention from researchers as a promising and practical predictive modeling approach based on the statistical learning theory. (For further details on these theoretical advances, see [11-14]).

3.1 Computational and statistical learning theory

A model generation process can be viewed as selecting a “best” possible model, from a given family of models, i.e. functions that map input feature space to the target variable. The models summarize data, examples of input/output combinations, that are a random sample of a given concept class. A model is “best” if it optimizes the error rate or more generally a loss function defined over the example space and the predicted output. Computational learning theory is concerned with the

complexity of such a process, but more focused on finding when the process can be efficient. One of the key areas in computational learning theory is the PAC, Probably Approximately Correct, learning model following the pioneering work of Valiant in the mid-80s. Informally speaking, a concept from a given concept class is PAC learnable if a model can be found from a given model class such that the “error” of the model on the examples of the concept is bound by some given ϵ within a given confidence bound of δ . The learning algorithm is said to be efficient if the complexity is polynomial in the number of examples needed to learn the concept, $1/\epsilon$ and $1/\delta$. One of the surprising results is that PAC learning of a concept from one class of concepts (e.g. learning a 3-term DNF) with models from the same class may be NP-hard, while learning with an expanded class of models (e.g. 3-term CNF formulae which includes the 3-DNF) is efficient. This implies that a judicious choice of the model class is important for practical model generation.

There are many negative results as to what is not PAC learnable. However, if there is a PAC learner with ϵ less than $1/2$, it can be turned into a strong learner with “error” close to 0, by multiple models as long as the weak model is for arbitrary example distribution, giving rise to a theoretical understanding of boosting (see section 4). Another interesting direction of the computational learning theory is to allow learning algorithm to make certain queries about the data vectors. By asking for the probability of the example event within some given “noise”-tolerance, this approach makes it possible to analyze the learning problems in the presence of noise, as needed for data mining. This line of research has also been applied to the minimal labeling problem of text mining with some promising results.

Statistical learning theory has its origin in the work of Vapnik and Chervonenkis in the late 60s, who developed a mathematical basis for comparing models of different forms. The theory focuses on finite sample statistics (classical statistics usually rely on asymptotic statistics) in the process of determining what is the best among the given set of models in fitting the data. In classical statistics approach, it is assumed that the correct model is known and the focus is on the parameter estimation. Statistical learning theory approach focuses on estimating relative performance of competing models so that the best can be selected.

For predictive models, consider an example vector \mathbf{z} given as the input feature vector \mathbf{x} and the target value y . A model α operates on \mathbf{x} and predicts $f(\mathbf{x}; \alpha)$. If we are modeling the conditional probability distribution of y as a function of \mathbf{x} , an appropriate loss function $Q(\mathbf{z}, \alpha)$ of the model on \mathbf{z} is the same negative log-likelihood often used in classical statistical modeling: $Q(\mathbf{z}, \alpha) = -\log p(y|\mathbf{x}; \alpha)$. For classification error, the loss function $Q(\mathbf{z}, \alpha) = 0$ if $y = f(\mathbf{x}; \alpha)$ and 1 otherwise. If it is only known that the data vector \mathbf{z} is generated according to some given probability measure $F(\mathbf{z})$, the best model would be the α that minimizes the expected loss

$$R(\alpha) = \int Q(\mathbf{z}, \alpha) dF(\mathbf{z}).$$

In practice, the probability measure $F(\mathbf{z})$ is not known, and one can only compute an empirical expected loss for the given example set $\mathbf{z}_i, i = 1, \dots, \ell$, assuming that they are *iid* generated:

$$R_{emp}(\alpha, \ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} Q(\mathbf{z}_i, \alpha).$$

Under what conditions does minimizing the empirical loss $R_{emp}(\alpha, \ell)$ also minimize the expected loss $R(\alpha)$ without knowing the distribution $F(\mathbf{z})$? Statistical learning theory answers this key question by offering the confidence regions for $R(\alpha)$ given $R_{emp}(\alpha, \ell)$ for a given lower bound of probability $1 - \eta$. These bounds are based on a measure known as VC–dimension of the set of models being considered. It suffices to say here that the VC–dimension can be considered to be a more reliable measure of the complexity of the model family than the degree of freedom concept used in the classical statistics. And it directly implies that the number of examples should be much more than the VC–dimension to obtain a reliable model.

While the VC–dimension of a set of models is often very difficult to compute, the theory does offer a practical method for selecting the “best” model, when there is “enough” data, by use of randomly split data set (into training and validation sets): the search for the best fitting model proceeds using the training set and the loss is estimated from the validation set using the confidence bound. The trick uses the fact that the confidence bound can be expressed without explicit dependence on the VC–dimension for the loss in the validation set. This is similar to the cross validation method in classical statistics where the technique is often used in searching for the parameter values of a fixed model family. The main difference here is that the search for the expected minimum-loss model using statistical learning theory allows for an arbitrary mixture of models.

Another important advance from the statistical learning theory is a new model family known as support vector machine described next.

3.2 Support Vector Machine

Support vector machine has the form of a linear discriminator. A linear discriminator can separate arbitrary partition of $d + 1$ points in d dimensions, hence the VC–dimension is at most $d + 1$ which is its degree of freedom in classical statistical sense. The number of input variables plus one and hence the number of coefficients to be determined for a linear discriminator, however, does not

mean the VC–dimension is the same number when the variables are not independent. Here we give a brief sketch of support vector machine model. (For a detailed introduction to the subject, see [14].) Let D be the smallest radius of the sphere that contains the data (example vectors). The points in either side have distances to the separating hyperplane. The smallest such distance is called the *margin* of separation. The hyper plane is called optimal if the margin is maximized. Let ρ be the margin of the optimal hyperplane. The points that are distance ρ away from the optimal hyperplane are called the *support vectors*. It has been shown that the VC–dimension is not greater than $\frac{D^2}{\rho^2} + 1$ and furthermore it is no greater than the number of support vectors. This is important because the VC–dimension of all possible separating hyperplanes for the data is shown to be not a direct function of the dimensionality of the vectors. This implies that one can generate arbitrarily many derived features, e.g. all pairwise products of the original features, as long as the number of support vectors for the optimal hyperplane (in the expanded dimensions) does not increase much. One can see that, although the final form of the model is a linear function, the decision boundary can be of almost arbitrary shape in the original feature space because of the nonlinearity introduced by derived variables.

This understanding leads to a new strategy to search for the support vectors and the coefficients for the optimal hyperplane simultaneously as an optimization problem in the expanded dimensional space. Actually, one need not explicitly compute the values of the derived features if they are chosen in a judicious way. The feature expansion makes use of several popular family of kernel functions, such as polynomial, radial basis functions or sigmoid functions as in the two layer neural network. efficient search techniques for the optimal hyperplane and the support vectors simultaneously and selecting the right family of feature extension are active areas of research. The support vector machine is a significant new addition for predictive modeling.

4 Use of multiple models

Recent research results in learning demonstrate the effectiveness of combining multiple models of the same or different types for improving modeling accuracy. These methods, variously termed as bagging [15], boosting [16], and stacking [17], have taken different approaches to achieve maximized modeling performance.

Let's look at an example where diverse predictive methods can be applied to obtain a solution. For example, a classification problem that can be solved by either a neural net method or a decision tree or a linear method. Until recently, the typical approach would be to try both methods on the same data, and then select the method with the strongest predictive performance. Stacking offers

an alternative approach that combines different models of solutions. In this example, the neural net, the decision tree and the linear solutions are found independently. For a new case, the answers from all methods are weighted and voted, for example 50% neural net and 25% each for the other methods. How are the appropriate percentages found? They can be readily learned by preparing a new dataset of derived cases, where the features are the answers give by each method and the correct answer is the case label.

Researchers have observed that predictive performance often can be improved by inducing multiple solutions of the same type, for example multiple decision trees. These models are generated by sampling from the same data [15]. The final answer on a new case is determined by giving a vote to each of the multiple decision trees, and picking the answer with the most votes.

Although the techniques for generating multiple models from the same data are independent of the type of model, the decision tree is the most commonly used. How are the trees generated? No change is made to a standard tree induction method. The difference is in the sampling method. In the simplest approach, called bagging [15], a sample of size n is taken with replacement from the original set of n examples. (For very large data, a smaller sample can be taken.) Some examples will be repeated in the sample, others may not occur. The expected proportion of unique examples in any given sample is 63.2%. Thus, it is possible to generate many samples, induce a decision tree from each sample, and then vote the results.

An alternative method of sampling, adaptive resampling, usually performs better than bagging. Instead of sampling all cases randomly, so that each document has a $1/n$ chance of being drawn from the sample, an incremental approach is used in random selection. The objective is to increase the odds of sampling documents that have been erroneously classified by the trees that have previously been induced. Some algorithms use weighted voting, where some trees may be given more weight than others. The “boosting” algorithm, such as AdaBoost [16] uses weighted voting and an explicit formula for updating the likelihood of sampling each case in the training sample. However, it has been demonstrated that unweighted voting and resampling with any technique that greatly increases the likelihood of including erroneous cases is also effective in greatly improving accuracy [15].

Each model has its own bias and variance component of errors. Multiple models of the same kind, e.g. trees, mainly reduce the variance portion of the error, while a mixture of models can reduce the bias effects in participating models as well. Combining multiple models gain primarily by countering the inadequacy of the fixed shape of the decision surface one model in the given family can produce and also by mitigating the inadequacy of search techniques associated with all practical model generation processes. While an ensemble models does not necessarily increase the

complexity of the model (in the sense of statistical learning theory), such a combined model does deny what understandability that might have existed in a single model. There are many other ways an ensemble of base models can be employed [18].

5 Feature selection and discretization

In some applications, there may be hundreds of features, most of which are irrelevant to the modeling target variable or are redundant. The excess features not only burden the computational process of generating a model, but they also interfere with the heuristic search for all practical model generation algorithms. Many algorithms will produce an inferior model in the presence of many irrelevant features. Thus the judicious selection of relevant set of features has been a major concern for the predictive modeling process. Stepwise regression or selectively removing the features with small coefficients in least squares fitting has been widely used in statistical modeling. Using a feature’s strength of correlation to the target variable (Gini index, information measures, Chi-square tests, etc) for either selecting or discarding the features has also been used in more recent family of models such as decision trees, rules and neural networks. Although these techniques are generally useful, they fail to select a near optimal set of features when the “correlation” is conditionally dependent on other feature’s values. So all methods that determine a feature’s merit based on some form of “correlation” measure are myopic. And all discretization of numerical feature values, whether implicit as in a decision tree, or explicit by preprocessing the data are myopic if the measures used in determining the interval boundaries are based on some myopic measure. We shall discuss two basic approaches to overcome the myopia in feature selection.

One approach is to *generate* a feature subset by some heuristic and *test* by actually building the model with the subset and evaluating its performance. Heuristics are necessary because it is not practical to test exponentially many unique subsets. Some feature selection techniques use genetic algorithms for this purpose [19]. The wrapper technique [20] and its follow on techniques either grow or shrink the feature subset by one “best” feature at a time, using multi-fold cross validation (building the model many times per candidate) for determining the performance due to each candidate feature. The iteration stops when no candidate improves the performance. This approach selects features specific to a chosen model family, for example a tree or a neural network.

The other approach can be characterized by the degree of importance or merit of a feature: by considering the local contexts in which a feature “correlates” to the target variable. This approach requires a definition of distance between the examples in the feature space. Let $d_{r,s,i}$ denote the i -th feature value distance between the two examples r and s . Without discussing the details,

this quantity ranges between 0 and 1 for numerical features, and has the value 0 if the values are same, or 1 if they are different for nominal features. The manhattan distance D_{rs} between the two examples r and s is just the sum of these component distances for all features: $D_{rs} = \sum d_{rs,i}$. Once the distance is defined we can determine the examples that are nearest to a given example. Let $S(r, k)$ denote the set of k -nearest examples of the same class to a given example r and $T(r, k)$ those of different class.

For brevity, we shall consider just two class situation here. The RELIEF family of feature merit measures are developed under the rationale that a good feature must be able to discriminate, by its value difference, close pairs of different class examples, and it is desirable to have the same value for the close pairs of the same class. The RELIEF merit $RM(X_i)$ for the feature variable X_i can be expressed within some constant factor as:

$$RM(X_i) = \sum_r \left(\sum_{s \in T(r, k)} d_{rs,i} - \sum_{s \in S(r, k)} d_{rs,i} \right),$$

where the examples for the outer sum may be taken from some random subset of the examples to save computation when there are abundance of examples. Kira and Rendell's original RELIEF [21] used single nearest neighbor, i.e., $k = 1$. Kononenko [22] and follow-on work [23,24] have generalized this basic measure for multiple class case, regression, and for robustness by using k -nearest neighbors with some reasonable k (say, 10).

The contextual merit measure developed by Hong [25] is based on a different rationale. For each example r the focus is on the k -nearest examples of different class and the features that contribute to discrimination in that context are credited. Given such an (r, s) pair, first there is the notion of the strength of the context, say $1/D_{rs}$, which is shared among the distance contributing features in proportion to their component distance, $d_{rs,i}/D_{rs}$. The contextual merit $CM(X_i)$ is:

$$CM(X_i) = \sum_r \sum_{s \in T(r, k)} \frac{d_{rs,i}}{D_{rs}^2}.$$

Both of these context based non-myopic measures are used for feature selection and discretization of numeric features (so as to “maximize” the resultant merit of the discretized feature, see [25,26]). They are effective even when the features strongly interact. For details and discussions on the strength of these techniques see [27].

6 Scaling up for large data

One of the most pressing challenges of data mining in many application domains is the problem of scaling the model generation algorithms to handle a huge volume of data. Besides traditional sampling techniques, there are numerous new techniques developed in the past decade to address this challenge. In addition to predictive accuracy, efficiency is of paramount concern: computational complexity, data structures, disk-based algorithms, and the degree of interface with database management systems, parallel and distributed algorithms and optimization techniques all play a role in efficient implementations. For many critical applications, relying on some “quick and dirty” (linear in the number of examples) algorithms may not produce a quality predictive model.

We have already discussed the JAM [6] approach to use distributed mining for the fraud detection application. Multiple model techniques can be employed in a similar manner. There are many examples of parallel implementation of tree generation processes. The SPRINT approach [4], organizes each feature values with their ID and class values in separate disk files for the purpose of rapidly obtaining the necessary contingency tables for the impurity computations. SPRINT develops a GINI-index based, MDL pruned decision tree. As the nodes are split, the split variable’s value file is also split into left and right parts, and a hash table is built to quickly decide which branch an example falls through. Experiments have shown that this heavily disk based parallel approach can handle multi millions of examples very efficiently. Other approaches, that embed most of tree generation functions within a DBMS proper, have also been reported.

7 Practical performance measures

Predictive models are usually developed for some application purpose in mind. Although the accuracy measure is generally useful for evaluating the models, the utility of the model’s output is a more direct goal. Developing models to optimize the utility has been studied for long time, for example users can enter a utility function to the CART [2] decision tree induction program. When the utility of a prediction can be computed in terms of the prediction statistics, it can be used in the model generation process in many different ways, e.g. in determining a split in a tree, in pruning process, etc. When the utility is not easily expressed in terms of computable measures with the model generation process, the negative log loss function is often more useful than the accuracy measure.

Two ways of evaluating model’s performance, *ROC curves* and *lift curves* are of interest. These are not new, but can offer insight into how models will perform for many application situations.

Many classification models can be modified so that the output is a probability of the given class, and hence depending on the threshold or some other decision making parameter value, one can get a family of models from one, for example a tree or a rule set. The Receiver Operating Characteristic (ROC) curve originated from signal detection theory. It plots the true positive rate (y-axis) against the false positive rate (x-axis). If there are two models in this space one can obtain any performance on the connecting line of the two just by randomly using the models with some probability in proportion to the desired position on the line. This curve allows one to select the optimal model depending on the assumed class distribution at the time of prediction. Also any model that falls below the convex hull of other models can be ignored [28]. Figure 1 shows an example of a ROC curve and a lift curve.

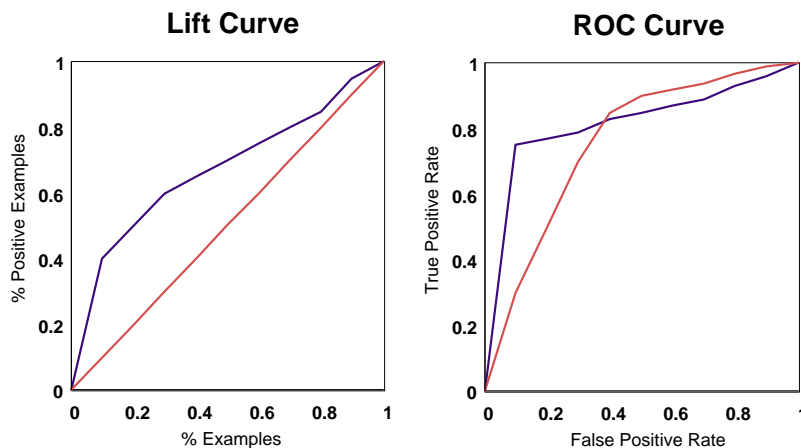


Figure 1: Lift and ROC Curves

For many applications, the aim of prediction is to identify some desired class members (e.g. customers) to whom some action (e.g. mailing advertisement circulars) is to be performed. Rather than classification, it is more flexible if the prediction is in the form of ranking. The probability of a given class, as in the ROC curve case, can be used for the ranking purpose. The Lift curve then plots cumulative true positive coverage in % (y-axis) against the ordered number of examples in % (x-axis). A random ranking will result in a straight diagonal line on this plot. A lift curve of a model is usually above this line, the higher the better. Some models with lower accuracy can have the lift curve above, on the average, a more accurate model, especially in the region of interest in the coverage (x-axis).

8 Conclusion

We have presented a brief picture of some notable advances in predictive modeling. There are many other advances of various significance. We will briefly mention just a few more. One of the areas that made much progress is the use of Bayesian networks for prediction purpose. There are new techniques to generate the network from the data itself. A surprising finding is that a simple, naive Bayesian classifier is often very effective and some simple extensions can make them even better [29]. New techniques are being developed that derive understandable rules from neural networks. Some progress is also made in finding useful hyperplanes as derived features to be used in generating trees, which can be especially useful for regression problems. Much work is continuing to generate probabilistic rules [5] that are tailored to the known distribution function of the target variable.

Many advances are ahead of us, and the research continues!

Acknowledgment

The authors are grateful for useful discussions and help in preparing this manuscript to their colleagues in the DAR group at IBM T.J. Watson Research Center. We express our special thanks to Edwin Pednault and Jonathan Hosking.

References

- [1] Gallagher C., "Risk Classification Aided by New Software Tool (CHAID -Chi-Squared Automatic Interaction Detector)", *National Underwriter Property & Casualty - Risk and Benefits Management*, Vol. 17, No. 19, April 1992.
- [2] Breiman L., Friedman J.H., Olshen R.A. & Stone C.J., *Classification and Regression Trees*, Wadsworth International Group, 1984.
- [3] Quinlan J.R., *C4.5 programs for machine learning*, Morgan Kaufmann, 1993.
- [4] Shafer J., Agrawal R, Mehta M., "SPRINT: A Scalable Parallel Classifier for data Mining", *Proc. of the 22nd ICVLDB*, pp. 544-555, 1996.
- [5] Apte C., Grossman E., Pednault E., Rosen B., Tipu F., White B, "Insurance Risk Modeling Using Data Mining Technology", *Research Report RC21314*, IBM Research Division, 1998. To appear in *Proc. of PADD99*.

- [6] Stolfo S.J., Prodromidis A., Tselepis S., Lee W., Fan W. & Chan P., “JAM: Java Agents for Meta-Learning over Distributed Databases”, *Proc. of KDDM97*, pp. 74-81, 1997.
- [7] Chan P.K. & Stolfo S.J., “Learning Arbiter and Combiner Trees from Partitioned Data for Scaling Machine Learning”, *Proc. of KDDM95*, pp. 39-44, 1995.
- [8] Hayes P.J. & Weinstein S., “Adding Value to Financial News by Computer”, *Proc. of the First International Conference on Artificial Intelligence Applications on Wall Street*, pp. 2-8, 1991.
- [9] Hayes P.J., Andersen P.M., Nirenburg I.B., & Schmandt L.M., “TCS: A Shell for Content-Based Text Categorization”, *Proc. of the Sixth IEEE CAIA*, pp. 320-326, 1990.
- [10] Weiss S. & Indurkha N., *Predictive Data Mining: A Practical Guide*, Morgan Kaufmann, 1998.
- [11] Hosking J.R.M., Pednault E.P.D. & Sudan M., “A Statistical Perspective on Data Mining”, *Future Generation Computer Systems: Special issue on Data Mining*, Vol. 3, Nos. 2-3, pp. 117-134, 1997.
- [12] Kearns M.J. & Vazirani U.V., *An Introduction to Computational Learning Theory*, MIT Press, 1994.
- [13] Mansour Y., “Lecture Notes on Learning Theory”, available from <http://www.math.tau.ac.il/~mansour>.
- [14] Vapnik V.N., *Statistical Learning Theory*, Wiley, 1998.
- [15] Breiman L., “Bagging Predictors”, *Machine Learning*, Vol. 24, pp. 123-140, 1996.
- [16] Freund Y. & Schapire R., “Experiments with a New Boosting Algorithm”, *Proc. of the International Machine Learning Conference*, Morgan Kaufmann, pp. 148-156, 1996.
- [17] Wolpert D., “Stacked Generalization”, *Neural Networks*, Vol. 5, No. 2, pp. 241-260, 1992.
- [18] Dietterich, T.D., “Machine learning Research: Four Current Directions”, *AI Magazine*, Vol. 18, No. 4, pp. 97-136, 1997.
- [19] Cherkauer K.J. & Shavlik J.W., “Growing Simpler Decision Trees to Facilitate knowledge Discovery”, *Proc. KDDM96*, pp. 315-318, 1996.
- [20] John G., Kohavi R. & Pfleger K., “Irrelevant features and the subset selection problem”, *Proc. 11th International Conf. on Machine Learning, ICML-94*, pp. 121-129, 1994.
- [21] Kira K. & Rendell L., “A practical approach to feature selection”, *Proc. Int. Conf. on Machine Learning, ICML-92*, pp. 249-256, 1992.

- [22] Kononenko I., “Estimating attributes: Analysis and extensions of RELIEF”, *Proc. European Conf. on Machine Learning*, pp. 171-182, 1994.
- [23] Kononenko I., Šimec E. & Robnik M., “Overcoming the Myopia of inductive learning algorithms with ReliefF”, *Applied Intelligence*, Vol. 7, pp. 39-55, 1997.
- [24] Robnik-Šikonja M. & Kononenko I., “An adaptation of Relief for attribute estimation in regression”, *Proc. Int. Conf. on Machine Learning, ICML-97*, Nashville, July 1997.
- [25] Hong S.J., “Use of contextual information for feature ranking and discretization”, *IBM Research Report RC19664*, July, 1994, a revised version to appear in *IEEE-TKDE*, Vol. 9, No. 4, 1997.
- [26] Robnik M. & Kononenko I., “Discretization of continuous attributes using ReliefF”, *Proc. Electr. and Comp. Sci. Conf., ERK-95*, pp. 149-152, 1995.
- [27] Kononenko I. & Hong S.J., “Attribute Selection for Modeling”, *Future Generation Computer Systems*, Vol. 13, Nos. 2-3, pp. 181-195, 1997.
- [28] Provost F., Fawcett T., & Kohavi R., “The Case Against Accuracy Estimation for Comparing Induction Algorithms”, *KDDM98*, 1998.
- [29] Domingos P. & Pazzani M., “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss”, *Machine Learning*, Vol. 29, pp. 103-130, 1997.