Mining databases with different schemas: Integrating incompatible classifers

Andreas L. Prodromidis [*] and **Salvatore Stolfo**
Department of Computer Science
Columbia University
1214 Amsterdam Ave. Mail Code 0401
New York, NY 10027
{andreas,sal}@cs.columbia.edu

Abstract

Distributed data mining systems aim to discover (and combine) usefull information that is distributed across multiple databases. The JAM system, for example, applies machine learning algorithms to compute models over distributed data sets and employs meta-learning techniques to combine the multiple models. Occasionally, however, these models (or classifiers) are induced from databases that have (moderately) different schemas and hence are incompatible. In this paper, we investigate the problem of combining multiple models computed over distributed data sets with different schemas. Through experiments performed on actual credit card data provided by two different financial institutions, we evaluate the effectiveness of the proposed approaches and demonstrate their potential utility.

Introduction

One of the main challenges in knowledge discovery and data mining is the development of systems capable of finding, analyzing and exploiting useful information that is distributed across several databases.

One approach of mining and combining such information is to apply various machine learning programs to discover patterns exhibited in the data and then combine the computed descriptive representations. Combining multiple classification models has been receiving increased attention(Dietterich 1997). Much of the work, however, is concerned with combining models obtained from different subsets (not necessarily distinct) of a single data set as a means to increase accuracy, (e.g. by imposing probability distributions over the instances of the training set, or by stratified sampling, sub-sampling, etc.) and not as a means to integrate distributed information. Although the *JAM* system (Stolfo *et al.* 1997) addresses the later by employing meta-learning techniques, integrating classification models derived from distinct and distributed databases may not always be feasible.

In all cases considered so far, all classification models are assumed to originate from databases of identical schemas. Since classifiers depend directly on the format of the underlying data, minor differences in the schemas between databases derive incompatible classifiers, yet these classifiers may target the same concept. Assume, for instance, two data sets of credit card transactions (labeled fraud or legitimate) from two different financial institutions (i.e. banks). The learning problem is to distinguish legitimate from fraudulent use of a credit card. It is desirable for both institutions to exchange their classifiers and hence incorporate in their system useful information that would otherwise be inaccessible to both. Indeed, for each credit card transaction, both institutions record similar information, however, they also include specific fields containing important information that each has acquired separately and which provides predictive value in determining fraudulent transaction patterns. In a different scenario where databases and schemas evolve over time, it may be desirable for a single institution to combine classifiers from both past accumulated data with newly acquired data.

Integrating the information captured by such classifiers is a non-trivial problem that we call, *"the incompatible schema"* problem. (The reader is advised not to confuse this with Schema Integration over Federated/Mediated Databases.) In this paper, first we formulate this problem and then we detail several techniques that "bridge" the schema differences and allow *JAM* and other data mining systems to share *incompatible* and otherwise useless classifiers. In the last part of the paper, we describe our experiments on credit card data obtained by two independent institutions and we establish that combining models from different sources can substantially improve performance.

Database Compatibility

Meta-learning (Chan & Stolfo 1993) is a technique that addresses the scaling problem of machine learning, i.e. the problem of learning useful information from large and inherently distributed databases. The idea is to execute a number of concept learning processes on a num-

[*]Supported in part by an IBM fellowship

This research is supported by the Intrusion Detection Program (BAA9603) from DARPA (F30602-96-1-0311), NSF (IRI-96-32225, CDA-96-25374) and NYSSTF (423115-445).

ber of data subsets in parallel, and combine their collective results through another phase of learning. Initially, each concept learning task or *base learner*, computes a concept or *base classifier* that models its underlying data subset or *training set*. Next, a separate concept learning task, called the *meta learner*, combines these independently computed base classifiers into a higher level concept or classifier, called a *meta classifier*, by learning from a *meta-level training set*. This meta-level training set is basically composed from the predictions of the individual base-classifiers when tested against a separate subset of the training data, also called a *validation set*. From their predictions, the meta-learner will detect the properties, the behavior and performance of the base-classifiers and compute a meta-classifier that represents a model of the "global" data set.

The *JAM* system (Stolfo *et al.* 1997) is designed to implement meta-learning. Briefly, *JAM* is a distributed agent based data mining system that provides a set of learning agents that compute models (concepts) over data stored locally at a site and a set of *meta-learning* agents for combining multiple models that were learned (perhaps) at different sites. The system employs a special distribution mechanism which allows the migration of the derived models or *classifier agents* to other remote sites. The *"incompatible schema"* problem, however, impedes *JAM* from taking advantage of all available databases.

Problem description Lets consider two data sites A and B with databases DB_A and DB_B respectively with similar but not identical schemas. Without loss of generality, we assume that:

$$Schema(DB_A) = \{A_1, A_2, ..., A_n, A_{n+1}, C\} \quad (1)$$
$$Schema(DB_B) = \{B_1, B_2, ..., B_n, B_{n+1}, C\} \quad (2)$$

where, A_i, B_i denote the $i-th$ attribute of DB_A and DB_B, respectively, and C the class label (e.g. the fraud/legitimate label in the credit card fraud example) of each instance. Without loss of generality, we further assume that $A_i = B_i$, $1 \leq i \leq n$. As for the A_{n+1} and B_{n+1} attributes, there are two possibilities:

1. $A_{n+1} \neq B_{n+1}$: The two attributes are of entirely different types drawn from distinct domains. The problem can then be reduced to two dual problems where one database has one more attribute than the other, i.e.:

$$Schema(DB_A) = \{A_1, A_2, ..., A_n, A_{n+1}, C\} \quad (3)$$
$$Schema(DB_B) = \{B_1, B_2, ..., B_n, C\} \quad (4)$$

where we assume that B_{n+1} is not present in DB_B.[1]

2. $A_{n+1} \approx B_{n+1}$: The two attributes are of similar type but slightly different semantics that is, there may be a map from the domain of one type to the domain of the other. For example, A_{n+1} and B_{n+1} are fields with time dependent information but of different duration (i.e. A_{n+1} may denote the number of times an event occurred within a window of half an hour and B_{n+1} may denote the number of times the same event occurred but within ten minutes).

[1]The dual problem has DB_B composed with B_{n+1} but A_{n+1} is not available to A.

In both cases the classifiers C_{Aj} derived from DB_A are not compatible with DB_B's data and hence cannot be directly used in DB_B's site, and vice versa. In the next section we investigate ways, called *bridging* methods, that overcome this incompatibility problem

Bridging Methods

There are several approaches to address the problem depending upon the learning task and the characteristics of the different or missing attribute A_{n+1} of DB_B:

- A_{n+1} **is missing, but can be predicted:** It may be possible to create an auxiliary classifier, which we call a *bridging agent*, from DB_A that can predict the value of the A_{n+1} attribute. To be more specific, by deploying regression methods (e.g. Cart (Breiman *et al.* 1984), locally weighted regression (C. G. Atkeson 1997), linear regression fit (Myers 1986), MARS (J.H.Friedman 1991)) for continuous attributes and machine learning algorithms for categorical attributes, data site A can compute one or more auxiliary classifier agents C_{Aj}' that predict the value of attribute A_{n+1} based on the common attributes $A_1, ..., An$. Then it can send all its local (base and bridging) classifiers to data site B. At the other side, data site B can deploy the auxiliary classifiers C_{Aj}' to estimate the values of the missing attribute A_{n+1} and present to classifiers C_{Aj} a new database DB_B' with schema $\{A_1, ..., A_n, \hat{A}_{n+1}\}$. From this point on, meta-learning and meta-classifying proceeds normally.
- A_{n+1} **is missing and cannot be predicted:** Computing a model for a missing attribute assumes a correlation between that attribute and the rest. Nevertheless, such a hypothesis may be unwarranted in which case we adopt one of the following strategies:
  - **Classifier agent C_{Aj} supports missing values:** If the classifier agent C_{Aj} originating from DB_A can handle attributes with missing values, data site B can simply include null values in a fictitious A_{n+1} attribute added to DB_B. The resulting DB_B' database is a database compatible with the C_{Aj} classifiers. Different classifier agents treat missing values in different ways. Some machine learning algorithms, for instance, treat them as a separate category, others replace them with the average or most frequent value, while most sophisticated algorithms treat them as "wild cards" and predict the most likely class of all possible, based on the other attribute-value pairs that are known.
  - **Learning agents at data site B can not handle missing values:** If, on the other hand, the classifier agent C_{Aj} cannot deal with missing values, data site A can learn two separate classifiers, one over the original database DB_A and one over

$DB_A{}'$, where $DB_A{}'$ is the DB_A database but without the A_{n+1} attribute:

$$DB_A{}' = PROJECT\ (A_1, ..., A_n)\ FROM\ DB_A \quad (5)$$

The first classifier can be stored locally for later use by the local meta-learning agents, while the later can be sent to data site B. Learning a second classifier without the A_{n+1} attribute, or in general with attributes that belong to the intersection of the attributes of the databases of the two data sites, implies that the second classifier makes use of only the attributes that are common among the participating data sites. Even though the rest of the attributes may have high predictive value for the data site that uses them, they are of no value for the other data site since they were not included anyway.

- A_{n+1} **is present, but semantically different:** It may be possible to integrate human expert knowledge and introduce *bridging* agents either from data site A, or data site B that can preprocess the A_{n+1} values and translate them according to the A_{n+1} semantics. In the context of the example described earlier where the A_{n+1} and B_{n+1} fields capture time dependent information, the bridging agent may be able to map the B_{n+1} values into A_{n+1} semantics and present these new values to the C_{Aj} classifier. For example, the agent may estimate the number of times the event would occur in thirty minutes by tripling the B_{n+1} values or by employing more sophisticated approximation formulas using non uniformly distributed probabilities (e.g. Poisson).

Experiments and Evaluation

To evaluate these techniques, we used 5 inductive learning algorithms (ID3, C4.5, Cart (Breiman *et al.* 1984), Naive Bayes (Minksy & Papert 1969) and Ripper (Cohen 1995)) and 2 data sets of real credit card transactions provided by the Chase and First Union Banks.

Each bank supplied .5 million records spanning one year. Chase bank records were sampled uniformly with 20% fraud versus 80% non-fraud distribution, whereas First Union data were sampled in a non-uniform manner (many records from some months, very few from others) with 15% fraud versus 85% non-fraud. Although both sets consist of credit card transactions prelabeled as fraudulent or legitimate, the schemas of the databases were developed separately and each data set also includes special features. Here, we demonstrate that these techniques enable the exchange of fraud detectors between the two banks in the hope of catching more fraud that either bank would be able to do alone.

To evaluate and compare the base- and meta-classifiers constructed, we adopted three metrics: the overall accuracy, the $(TP - FP)$ spread and a cost model fitted to the credit card detection problem. Overall accuracy expresses the ability of a classifier to give correct predictions, $(TP - FP)$ denotes the ability of a classifier to catch fraudulent transactions while minimizing false alarms, and finally, the cost model captures the performance of a classifier with respect to the goal of the target application (stop loss due to fraud).

Credit card companies have a fixed overhead that serves as a threshold value for challenging the legitimacy of a credit card transaction. In other words, if the transaction amount *amt*, is below this threshold, they choose to authorize the transaction automatically. Each transaction predicted as fraudulent require an “overhead” referral fee for authorization personnel to decide the final disposition. This “overhead” cost is typically a “fixed fee” that we call $\$X$. Therefore, even if we could accurately predict and identify all fraudulent transactions, those whose *amt* is less than $\$X$ would produce $(X - amt)$ in losses anyway. In these experiments, we incorporate the threashold values and referral fees in the detection process and we seek to produce classifiers and meta-classifiers that maximize the total savings.

The two databases have the following differences:

1. Chase includes two (continuous) features not present in the First Union data
2. Chase and First Union defines a (nearly identical) feature with different semantics

For the **first** incompatibility, we extend the First Union data set with two additional fields padded with null values to fit it to the Chase schema and we deploy classifier agents that support missing values. For the **second** incompatibility, we have the values of the First Union data mapped to the semantics of the Chase data.

The results plotted are averaged over 6 fold cross validation. Figure 1 present the accuracy, the $(TP - FP)$ spread and the savings (in dollars) of the Chase (three upper graphs) and First Union meta-classifiers (three lower graphs) on Chase and First Union data, respectively. The first (Chase meta-classifiers) rely solely on First Union base classifiers while the later (First Union meta-classifiers) depend on Chase base-classifiers alone. The curves of each figure represent the performance of the 5 different meta-learning algorithms as they combine more base-classifiers. The order of the base-classifiers selected is determined by a special pruning algorithm that aims to improve the $(TP - FP)$ rate while maximizing coverage (i.e. the percentage of transactions classified correctly by at least one base-classifier) (Prodromidis, Stolfo, & Chan 1998).

The results demonstrate that both Chase and First Union classifiers can be exchanged and applied to each of their respective data sets and catch significant amount of fraud despite the fact that the base classifiers are trained over data sets with different characteristics, patterns and fraud distribution.

With two features missing from First Union data, Chase base classifiers are unable to take full advantage of their knowledge regarding credit card fraud detection. To alleviate the problem, we deployed regression techniques in an attempt to approximate the missing values for one of the missing features of the First Union

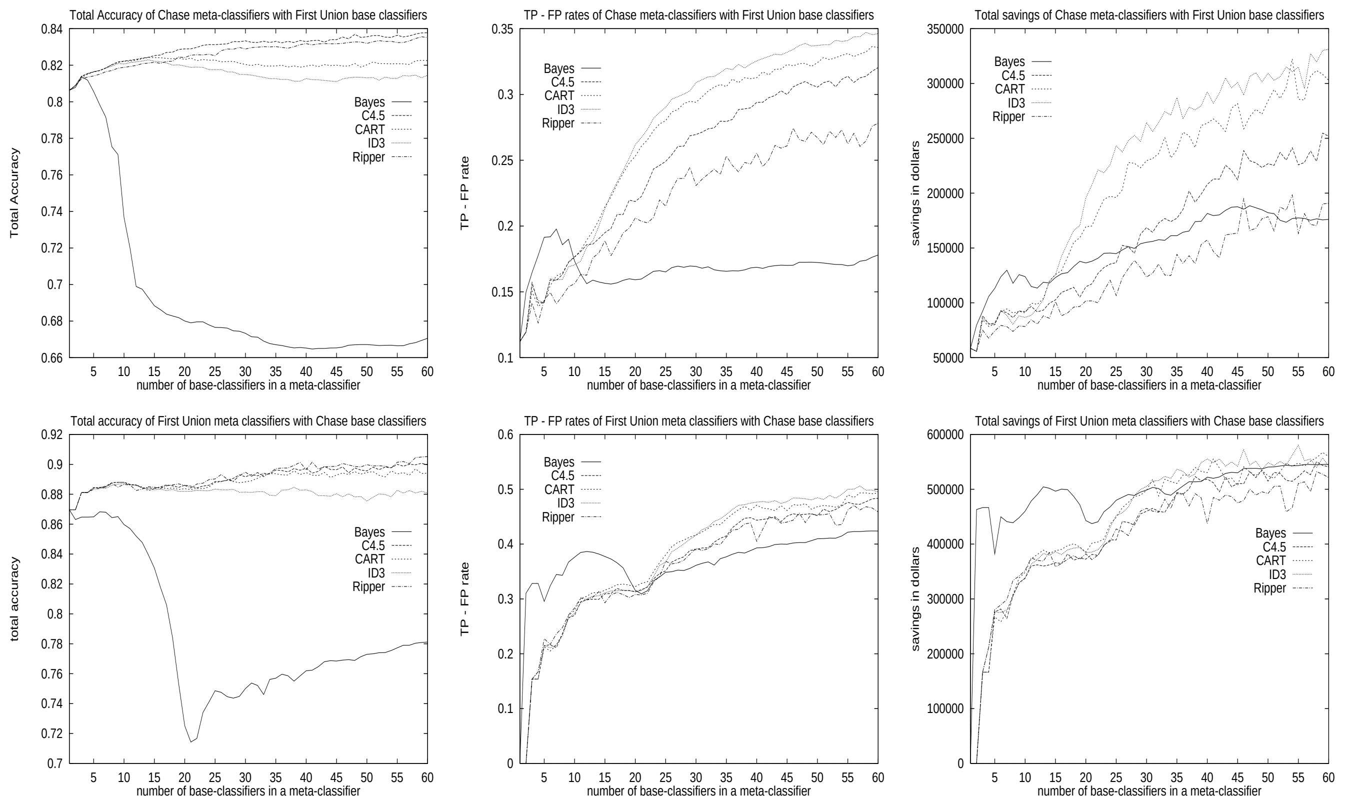

Figure 1: Accuracy (left), $TP - FP$ spread (middle), and savings (right) of Chase meta classifiers/First Union base classifiers (top) and First Union meta-classifiers/Chase base classifiers (bottom).

data. In this experiment, we tested two different regression methods, the linear regression fit and regression trees both available by Splus (Sta 1996). We applied the two techniques on the different subsets of the Chase data to generate several bridging models and we matched each (original) Chase classifier with one bridging classifier according to their performance on a First Union validation set. In Figure 2 (first row) we display the performance (accuracy, $TP - FP$ spread and total savings) of the Chase base-classifiers with (black bar) and without (grey bar) the assistance of these bridging models. The first 12 bars represent the classifiers derived by the Bayesian learning algorithm when trained against the 12 different months of data, followed by the C4.5, Cart, ID3 and Ripper learning algorithms.

The degree the base classifiers can benefit from bridging classifiers depends on the "importance" of the missing feature (e.g. its information gain or predictive value), the "dependence" of this feature to the remaining known attributes, the bias of the bridging learning algorithm and the quality of the training and testing data. In this case, as the figures demonstrate, the bridging classifiers are quite beneficial for the majority of the Chase base classifiers.

After being matched and evaluated, the base-classifier/bridging classifier pairs can be integrated into new meta-classifier hierarchies. In the middle row of Figure 2 we present the performance results of the new First Union meta-classifiers. By contrasting the accuracy (left), the $TP - FP$ spread (middle) and total savings (right) plots of this figure against the three bottom plots of Figure 1 we determine that the improved performance of the base-classifiers due to the bridging classifiers resulted in significantly superior meta-classifiers. This translates to 6.3% higher accuracy, 32.9% higher $TP - FP$ spread and \$284K in additional savings.

Finally the last row of Figure 2 displays the performance of the First Union meta-classifiers when combining all available base-classifiers, both, the local First Union base classifiers and the Chase base/bridging classifier pairs. The new First Union meta-classifier is superior even to the best "pure" First Union meta-classifiers (i.e. the meta-classifier composed by local base-classifiers alone) as reported in (Prodromidis, Stolfo, & Chan 1998) improving total accuracy by 1.5%, $TP - FP$ spread by 5.3% and savings by \$20K.

References

Breiman, L.; Friedman, J. H.; Olshen, R. A.; and Stone, C. J. 1984. *Classification and Regression Trees.* Belmont, CA: Wadsworth.

C. G. Atkeson, S. A. Schaal, A. W. M. 1997. Locally weighted learning. AI Review, To Appear.

Chan, P., and Stolfo, S. 1993. Meta-learning for mul-

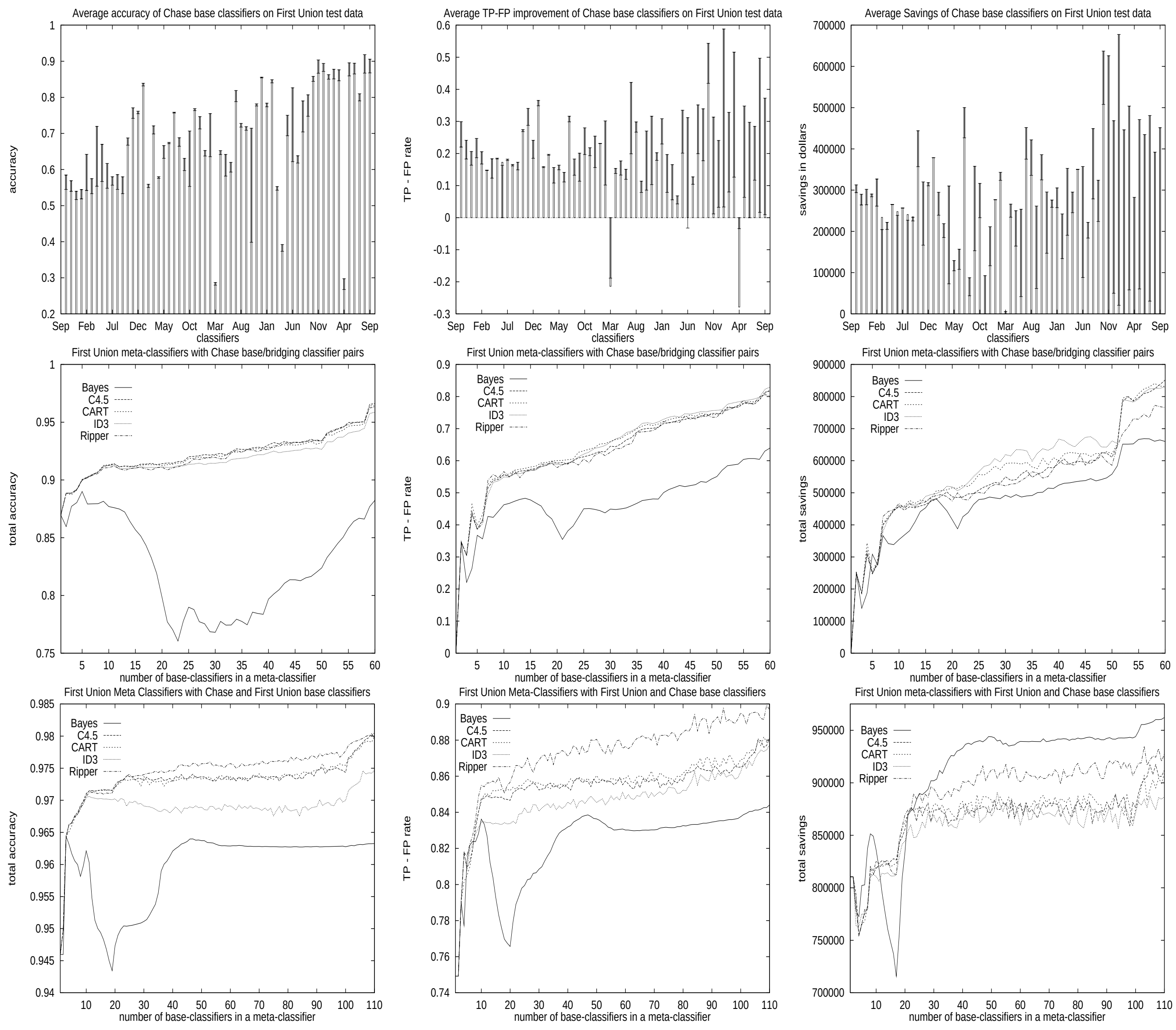

Figure 2: Top: Accuracy (feft), $TP - FP$ spread (middle) and total savings (right) of plain Chase base-classifiers (grey bars) and of Chase base-classifier/bridging classifier pairs (black bars) on First Union data. Center: Total accuracy (left), $TP - FP$ spread (middle) and total savings (right) of First Union meta-classifiers composed by Chase base-classifier/bridging-classifier pairs. Bottom: Total accuracy (left), $TP - FP$ spread (middle) and total savings (right) of First Union meta-classifiers combining both Chase and First Union base classifiers.

tistrategy and parallel learning. In *Proc. Second Intl. Work. Multistrategy Learning*, 150–165.

Cohen, W. 1995. Fast effective rule induction. In *Proc. 12th Intl. Conf. Machine Learning*, 115–123.

Dietterich, T. 1997. Machine learning research: Four current directions. *AI Magazine* 18(4):97–136.

J.H.Friedman. 1991. Multivariate adaptive regression splines. *The Annals of Statistics* 19(1):1–141.

Minksy, M., and Papert, S. 1969. *Perceptrons: An Introduction to Computation Geometry.* Cambridge, MA: MIT Press. (Expanded edition, 1988).

Myers, R. 1986. *Classical and Modern Regression with Applications.* Boston, MA: Duxbury.

Prodromidis, A.; Stolfo, S.; and Chan, P. 1998. Pruning classifiers in a distributed meta-learning system. Technical report, Columbia Univ. CUCS-011-98.

StatSci Division, MathSoft. 1996. *Splus, Version 3.4.*

Stolfo, S.; Prodromidis, A.; Tselepis, S.; Lee, W.; Fan, W.; and Chan, P. 1997. JAM: Java agents for meta-learning over distributed databases. In *Proc. 3rd Intl. Conf. Knowledge Discovery and Data Mining*, 74–81.