Predictive Learning in Two-way Datasets

Beau Piccart[1], Hendrik Blockeel[1,2], Andy Georges[3], Lieven Eeckhout[3]

[1]Katholieke Universiteit Leuven, Department of Computer Science
[2]Leiden Institute of Advanced Computer Science, Universiteit Leiden
[3]Universiteit Gent

Abstract. We introduce a new learning setting, called two-way predictive learning, as a special case of relational learning. We demonstrate that this learning setting has some properties that make an alternative learning approach, which we refer to as transposed learning, possible. We show experimentally that transposed learning can yield better results in multi-target learning settings.

1 Introduction

Imagine that we have a set of objects $A \subseteq \mathcal{A}$ and a set of objects $B \subseteq \mathcal{B}$, and the data set contains all pairs $(\mathbf{a}, \mathbf{b})$ with $\mathbf{a} \in A$ and $\mathbf{b} \in B$, with a label assigned to each pair. We call this a two-way dataset. We consider predictive learning in this setting.

Definition 1 (Two-way predictive learning). *The task of two-way predictive learning is defined as follows:* **Given:** *a data set D that consists of a set of objects $A \subseteq \mathcal{A}$ and $B \subseteq \mathcal{B}$ and for each combination of* $\mathbf{a}$ *and* $\mathbf{b}$ *a corresponding label $f(\mathbf{a}, \mathbf{b})$, that is, $D = \{(\mathbf{a}, \mathbf{b}, f(\mathbf{a}, \mathbf{b}))|\mathbf{a} \in A, \mathbf{b} \in B\}$ with $A \subseteq \mathcal{A}$ and $B \subseteq \mathcal{B}$,* **Find:** *the function f.*

The two-way setting is relevant for many applications. These include molecular biology microarray data, recommender systems, multi-target prediction [2], and the related problem of multi-task learning [4]. Several toy examples in statistical relational learning, such as the student-course-grade example [6], essentially describe a two-way prediction problem.

Two-way learning is an instance of relational learning [5]: we predict an attribute of a relation from information about the participating objects. Propositionalization (representing the data using a single table) would cause redundancy (each object from A and B is described multiple times) and loss of information (the case where two rows refer to the same $\mathbf{a}$ can no longer be distinguished from that where two rows describe different objects with the same attribute values). Yet it is special, in the sense that the relation is complete (each $\mathbf{a}$ is linked to each $\mathbf{b}$), so there is no information in the structure of the relation itself. In this paper, we discuss a few peculiarities of this setting.

Table 1. Overview of two-way learning settings. In the table, * means 'non-empty'.

$Attr(\mathcal{A})$	$Attr(\mathcal{B})$	$Attr(\mathcal{R}) \setminus \{T\}$	
$\emptyset$	$\emptyset$	$\emptyset$	bare two-way learning
*	$\emptyset$	$\emptyset$	single-decorated two-way learning
$\emptyset$	*	$\emptyset$	single-decorated two-way learning
*	*	$\emptyset$	double-decorated two-way learning
$\emptyset$/*	$\emptyset$/*	*	relational learning with deterministic background

2 Situating two-way learning

Consider the following relational learning context: we have two types of objects $\mathcal{A}$ and $\mathcal{B}$, and a relation $\mathcal{R}$ between them. The objects of type $\mathcal{A}$ have attributes A_i, $i = 1, \ldots, n_A$; the objects of type $\mathcal{B}$ have attributes B_i, $i = 1, \ldots, n_B$; and the tuples in $\mathcal{R}$ have attributes R_i, $i = 1, \ldots, n_R$ as well as a special attribute T called the target attribute. We denote the set of attributes of $\mathcal{A}$, $\mathcal{B}$, $\mathcal{R}$ as $Attr(\mathcal{A})$, $Attr(\mathcal{B})$, $Attr(\mathcal{R})$, and their respective extensions as A, B, R. The relation R is complete: there is a relationship between each $\mathbf{a} \in A$ and each $\mathbf{b} \in B$. Figure 1 summarizes this in an entity-relationship diagram.

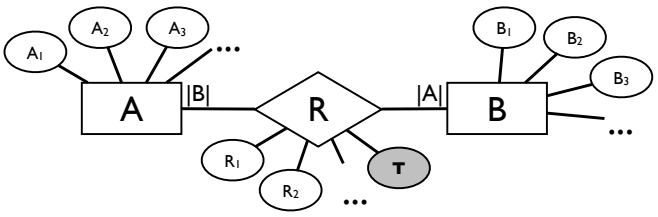

Fig. 1. ER-diagram summarizing the data types available for learning. T is the target attribute.

The task is to predict T from the other available information. We can consider multiple settings, depending on what attributes are available. Table 1 provides an overview. The settings covered by the ER-diagram can be called "relational learning with deterministic background" (since each instance of R is linked to exactly one A and B). Two-way learning, as defined here, covers the cases where $Attr(\mathcal{R}) = \{T\}$.

In the remainder of this paper, we focus on *bare two-way learning*. Since, in this setting, R is complete, $Attr(\mathcal{A}) = Attr(\mathcal{B}) = \emptyset$, and $Attr(\mathcal{R}) = \{T\}$, the data set D is simply a matrix. For each $\mathbf{a}_i$ and $\mathbf{b}_j$, we denote the corresponding T value as t_{ij}. This is the type of data we get in microarray data and in the context of recommender systems [1].

It may seem strange that we want to predict t_{ij} from no information at all, since $Attr(\mathcal{A}) = Attr(\mathcal{B}) = \emptyset$, but the point is that the values of T themselves

carry information. We can predict t_{ij} from the information in the $t_{ik}, k \neq j$, or from the $t_{kj}, k \neq i$, or even from the $t_{kl}, k \neq i, l \neq j$.

Bare two-way predictive learning can be addressed in different ways. Suppose we need to predict a single T_{ij} element. We distinguish the following main approaches:

Row-based: We learn a function f that predicts $T_{.j}$ from $T_{.k}, k \neq j$. That is, we reduce the task to a standard learning task, treating the rows as instances and the columns as attributes. The target attribute is $T_{.j}$, and the predictive attributes are $T_{.k}$ with $k \neq j$.

Column-based: This is the same as above, except that we treat the *columns* as instances and the *rows* as attributes; the target attribute is $T_{i.}$, and the predictive attributes are $T_{k.}$, $k \neq i$. We call this *transposed learning*, as it really corresponds to transposing the matrix that represents the data set and then using a standard learning method.

These two alternatives also exist in single-decorated and double-decorated two-way learning, as shown on Figure 2.

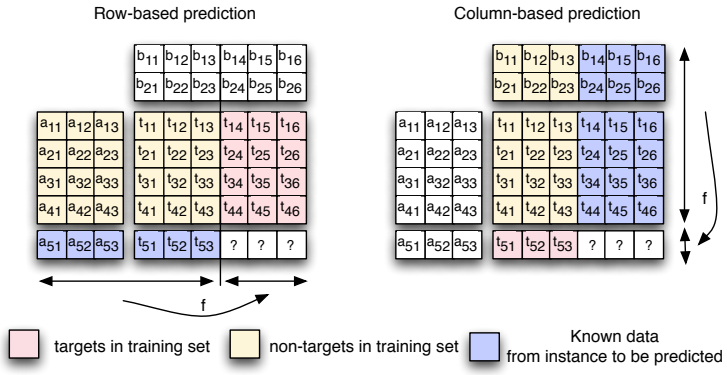

Fig. 2. We can learn a function f that generalizes over $\mathcal{A}$, predicting target values $t_{.i}$ from $a_{.j}$ and $t_{.k}$ or we can learn a function f that generalizes over $\mathcal{B}$, predicting target values $t_{i.}$ from $b_{j.}$ and $t_{k.}$. In the second case, the $t_{i.}$ targets will only become available when the new example becomes available.

3 The effects of transposition

In the two-way learning setting, rows and columns are to some extent interchangeable. Consider again Figure 2. Given a new object **a**, we need to predict the T_j values for it. First consider the user's point of view: rows are examples, columns are attributes. We get a new example and need to fill in some target values. If we have already learned a function f in the standard (row-based) way we can now apply it to predict those values.

Alternatively, we can consider a learner using the *transposed* view on the data. To this learner, **a** is a *new (target) attribute*. Since **a** is unknown during training, it is not possible to learn a function f till prediction time. At prediction time, f can be learned using the known components of **a** as known target values. In the column-based approach, our inductive learner is thus used lazily (or, transductively).

This transposition also turns a multi-target problem in to a single-target problem and vice versa. This can be seen in the example in Figure 2. The row-based approach results in a multi-target function f which predicts $(t_{.4}, t_{.5}, t_{.6})$ given $(a_{.1}, a_{.2}, a_{.3}, t_{.1}, t_{.2}, t_{.3})$ while the column-based approach results in a single-target function which predicts $t_{5.}$ given $(b_{1.}, b_{2.}, a_{1.}, t_{2.}, t_{3.}, t_{4.})$.

4 Applications

4.1 Microprocessor-data

For this application [8], the data consists of a set of performance numbers obtained by executing 26 benchmark programs[1] on a number of (different) machines. Based on these data, we wish to predict the performance of each of the machines for new programs. When a new program becomes available, the idea is to execute it on a limited number of machines ("predictive machines") and use this information together with the data about the benchmark programs to predict the performance of the remaining machines ("target machines"). Fig. 3 illustrates the task. Since we use no descriptors of the machines or the programs, besides the performance numbers, this is a bare two-way learning problem.

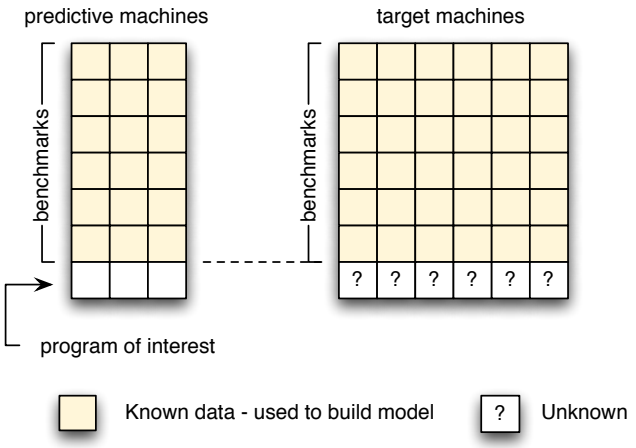

Fig. 3. Problem statement and terminology

[1] From the SPEC CPU20006 suite, `http://spec.org/cpu2006`.

We here compare row-based and column-based inductive learning. Our goal is to see whether the less natural column-based approach can offer an advantage over straightforward row-based learning.

We used three different machine learning algorithms from the Weka collection [7]: a multi-layer perceptron (MLP), a Support Vector Machine using Sequential Minimal Optimization (SVM-SMO), and linear regression (Linear Regression). Each algorithm was run both row-based and column-based. Default parameters were used for each algorithm and the performance is estimated using cross validation.

Table 2 shows that column-based prediction yields much better results than row-based prediction. One way to interpret this is that generalization over machines is easier than generalization over programs.

Table 2. Spearman Rank correlation for the predicted microprocessor data.

	Row-based	Column-based
Neural Network	0.82	**0.93**
SVM	0.77	**0.90**
Linear Regression	0.82	**0.91**

4.2 Ecological data

Next, we consider an ecological application [3]. Biologists take samples of river water to measure its quality, recording quantities of a number of micro-organisms, and physico-chemical parameters (such as oxygen concentration) in these samples. The goal is to learn to predict the physico-chemical parameters (PCP, for short) from the micro-organism quantities. This problem corresponds to a standard multi-target problem on which we can perform both row-based and column-based learning.

We use the same three algorithms as for the micro-processor data. Results are shown in Table 3. Here, column-based prediction outperforms row-based prediction for the MLP and SVM, while row-based prediction works better for linear regression. This indicates that the optimal learning direction can depend on the learning algorithm.

Table 3. Mean Squared Error (MSE) for the ecological dataset.

	Row-based	Column-based
Neural Network	1.127	**0.9**
SVM-SMO	2.43	**1.46**
Linear Regression	**3.07**	4.04

5 Conclusions

We propose a new setting for machine learning that we call *two-way predictive learning*. The setting is characterized by the existence of two types of data elements, pairs of which are labeled with target values to be predicted. The two-way learning setting is encountered in many application domains. It covers multi-target learning as a special case, and is itself a special case of relational learning. It has some pecularities that motivate a separate study of this setting. In particular, it raises interesting questions about the interchangeability of examples and attributes, which itself sheds new light on the difference between inductive and transductive learning.

Experiments on two different application domains demonstrate the usefulness of this discussion: by running an inductive learner on the transposed data matrix, one can obtain better predictive results. While this approach is practically very simple, it is not straightforward to practitioners, partially because it requires a view of the data that is often unnatural (thinking of attributes as examples and vice versa), and partially because it requires one to use an inductive learner transductively (the learning process can only be started once a new test instance has arrived). Our experimental results show, however, that this alternative approach can yield important performance gains.

Acknowledgement

The work was funded by the Research Foundation - Flanders (FWO-Vlaanderen), project G.0255.08, "Efficient microprocessor design using machine learning".

References

1. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
2. T. Aho, B. Zenko, and S. Dzeroski. Rule ensembles for multi-target regression. In Wei Wang, Hillol Kargupta, Sanjay Ranka, Philip S. Yu, and Xindong Wu, editors, *ICDM 2009, The Ninth IEEE International Conference on Data Mining, Miami, Florida, USA, 6-9 December 2009*, pages 21–30. IEEE Computer Society, 2009.
3. H. Blockeel, S. Dzeroski, and J. Grbovic. Simultaneous prediction of mulriple chemical parameters of river water quality with tilde. In *PKDD'99*, pages 32–40, 1999.
4. R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, July 1997.
5. L. De Raedt. *Logical and Relational Learning*. Springer, 2008.
6. L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. *Learning Probabilistic Relational Models*, pages 307–334. Springer, 2001.
7. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.
8. K. Hoste, A. Phansalkar, L. Eeckhout, A. Georges, L. K. John, and K. De Bosschere. Performance prediction based on inherent program similarity. In *Proceedings of the 15th international conference on Parallel architectures and compilation techniques*, PACT '06, pages 114–122, New York, NY, USA, 2006. ACM.