Cost-sensitive Parsimonious Linear Regression

Robby Goetschalckx Kurt Driessens
Katholieke Universiteit Leuven
Leuven, Belgium
Robby.Goetschalckx,Kurt.Driessens@cs.kuleuven.be

Scott Sanner
National ICT Australia
Canberra, Australia
Scott.Sanner@nicta.com.au

Abstract

We examine linear regression problems where some features may only be observable at a cost (e.g., in medical domains where features may correspond to diagnostic tests that take time and costs money). This can be important in the context of data mining, in order to obtain the best predictions from the data on a limited cost budget. We define a parsimonious *linear regression objective criterion that jointly minimizes prediction error and feature cost. We modify least angle regression algorithms commonly used for sparse linear regression to produce the ParLiR algorithm, which not only provides an efficient and parsimonious solution as we demonstrate empirically, but it also provides formal guarantees that we prove theoretically.*

1 Introduction

Linear regression models are some of the most widely used and well-studied models for regression. However, standard linear regression models often assume that all features can be evaluated without difficulty, while in many real-world regression scenarios this assumption does not hold. For example, in a medical domain where the objective is to regress the risk-level of a patient for some condition (quantified as a real-valued variable), features used in linear regression may correspond to the results of various diagnostic tests. Yet clearly, it may be impossible in a medical setting to evaluate all possible tests. A related problem may occur in a financial setting where a regression model of a stock price or the risk-level of a potential debtor often requires evaluating features corresponding to information-gathering actions that incur time and monetary costs. Clearly there is no benefit in reducing error by using more features if the financial advantage gained by obtaining more accurate predictions does not outweigh the financial cost of doing so.

These and related problems associated with the difficulty of evaluating features in a regression model can be formalized more precisely with the notion of feature costs. Assuming that prediction error and feature costs can be measured in commensurable units, we can reformulate our linear regression problem w.r.t. a *parsimonious*[1] objective criterion which jointly minimizes both prediction error and feature cost.

We examine efficient approximations to the parsimonious linear regression optimization problem. We are able to provide a modified least angle regression approach for parsimonious linear regression called ParLiR that encourages sparsity in the feature weights in a manner proportional to the feature cost. ParLiR runs in a time linear in the number of features and the amount of data, thus producing a tractable algorithm in contrast to direct optimization of the original objective criterion. As an empirical validation, we demonstrate the behavior of ParLiR on a number of UCI data-sets and show that ParLiR provides an efficient and parsimonious solution in comparison to other cost-aware linear regression approaches. We prove the important result that ParLiR ensures that every feature used in the regression reduces the error by at least its cost, thus proving parsimony. Additionally, in the special case where all features are orthogonal, we provide an outline for a proof that ParLiR does obtain the optimal solution.

2 Related Work

Turney [8] provides an overview of ways that *cost-sensitivity* may be introduced into machine learning as well as a comprehensive bibliographical reference to historical work in this area. With respect to the published literature, there are two main categories in which this work is concentrated:

- **Prediction error costs**: Here it is assumed that different prediction errors incur different penalties (see [2, 4, 6] for a sampling of recent work).
- **Feature evaluation costs:** This is the fundamental problem that we address. Historically, there has been

[1] Def. *Parsimonious* (adjective): Exhibiting the quality of being careful with money or resources.

work on the classification side of this problem [7]. Such approaches do not easily extend to the regression of continuously varying functions on a continuous input space. For such problems, linear regression models are commonly used although parsimonious extensions to deal with feature costs do not admit the simple modifications used for classification. As such, parsimonious extensions of linear regression to deal with feature costs as proposed in this paper comprise a novel contribution to machine learning.

Finally, we note that while we do make use of least angle regression methods related to sparse linear regression methods such as *lasso* [5], these algorithms by themselves do not guarantee parsimony w.r.t. feature costs. Furthermore, while the original least angle regression approaches solve a constrained quadratic programming problem, the corresponding parsimonious linear regression problem results in a constrained mixed 0-1 integer quadratic program formulation. As such, it is not clear that modified least angle regression techniques necessarily produce a low-cost solution to the parsimonious linear regression problem, nor one that closely approximates the global optimum. Thus, it is crucial to prove such properties for these modified algorithms in the parsimonious linear regression setting as we do in Section 5.

3 Parsimonious Linear Regression

We begin with the problem formulation for parsimonious linear regression and then proceed to specify the ParLiR algorithm that efficiently approximates its solution.

3.1 Problem Formulation

Our problem is identical to the linear regression setting with the added modification that features are costly. Formally, we are given the following information:

- A set of input features (i.e., variables) $\mathcal{X} = \{f_1, \ldots, f_n\}$ where each feature variable $f_i \in \mathbb{R}$.
- A finite cost c_i associated with each input feature f_i.
- A target response variable $y \in \mathbb{R}$.
- A set of m data samples $\mathcal{D} = \{\langle \mathcal{X}, y \rangle\}$ where we denote the particular assignment for data sample j ($1 \leq j \leq m$) in expanded form $\langle f_{1j}, \ldots, f_{nj}, y_j \rangle$.
- By affine transformations for each feature f_i, we assume the feature value vector $\vec{f_i} = \langle f_{i1}, \ldots, f_{im} \rangle$ is standardized to mean 0 and unit length.

Our objective is to find a linear regressor $\hat{y}(\mathcal{X}, \vec{w})$ w.r.t. data $\mathcal{D}$ and weight vector $\vec{w} = \langle w_0, \ldots w_n \rangle$ ($\vec{w} \in \mathbb{R}^{n+1}$) in the following linear form:

$$\hat{y}(\mathcal{X}, \vec{w}) = w_0 + \sum_{f_i \in \mathcal{X}} w_i f_i \tag{1}$$

If a weight $w_i = 0$, then we say that the feature f_i has not been selected. Formally, we define the subset of selected features for our regressor $\hat{y}$ with weights $\vec{w}$ as $\mathcal{F}_{\vec{w}} \subset \mathcal{X}$ such that $\mathcal{F}_{\vec{w}} = \{f_i \in \mathcal{X} | w_i \neq 0\}$. Then we can easily define the cost $C(\vec{w})$ of a particular selection of linear regression weights $\vec{w}$ using the following weighted $\mathcal{L}_0$ norm:

$$C(\vec{w}) = \sum_{f_i \in \mathcal{F}_{\vec{w}}} c_i. \tag{2}$$

We also define the usual average squared error function $E(\vec{w}, \mathcal{D})$ for weights $\vec{w}$ as the following

$$E(\vec{w}, \mathcal{D}) = \frac{1}{m} \sum_{\langle \mathcal{X}, y \rangle \in \mathcal{D}} \left(\hat{y}(\mathcal{X}, \vec{w}) - y \right)^2. \tag{3}$$

Assuming that prediction error and feature costs can be measured in commensurable units, we now reformulate our linear regression problem w.r.t. a *parsimonious* objective criterion — one that jointly minimizes both prediction error and feature cost.

Definition 3.1 (Parsimonious Linear Regression)**.** *Given input feature variables $\mathcal{X}$, target response variable y, and a target linear form $\hat{y}(\mathcal{X}, \vec{w})$ using weight variables $\vec{w}$, we define the parsimonious linear regression solution $\vec{w}$ to be a global optima of the following unconstrained optimization problem:*

$$\begin{aligned} &\text{Variables: } \vec{w} \\ &\text{Minimize: } C(\vec{w}) + E(\vec{w}, \mathcal{D}) \end{aligned}$$

Unfortunately, in this form, the above optimization objective is not convex in $\vec{w}$ – while the sum of squared error $E(\vec{w}, \mathcal{D})$ is well-known to be a convex function in $\vec{w}$ (i.e., it is the same error function used for ordinary least squares regression), $C(\vec{w}, \mathcal{D})$ has step discontinuities where any $w_i = 0$ and is therefore non-convex in $\vec{w}$, making the overall objective non-convex. Thus, in contrast to the case of ordinary least squares linear regression, we cannot apply unconstrained convex optimization techniques to directly solve this parsimonious linear regression problem. To find an optimal solution, one has to check the full power-set of the set of features, making the problem NP-complete.

Although the same approach is used in other work [1], one might object that it is unusual to consider squared error and cost in the same units. For example, if non-squared error and cost are both measured in the same units, then squared error cannot be directly traded off against non-squared cost. But this is not a problem; if the non-squared error and cost are in the same units then the squared cost may be used as the feature cost c_i so that the trade-off can be expressed in the same units. Because this is just a change of constants c_i, it requires no change to the algorithms.

3.2 Efficient Approximation

Due to their sparsity properties, which are useful for performing implicit feature selection, we focus on a class of linear regression techniques collectively referred to as *least-angle regression* (LAR) methods, such as *lasso* and *forward stage-wise regression* [3].

One of the key ideas behind least angle regression is that one may perform regression by incremental line search in single feature dimensions, specifically ordering feature dimensions by the amount they correlate with the regression error of the current solution. Furthermore, doing so often yields sparse solutions when there is a restrictive $\mathcal{L}_1$ constraint on the total sum of the weights. Least angle regression methods manage to closely approximate the optimal regression solutions to their respective problems formulated as quadratic programs [3].

For parsimonious linear regression, this single dimensional line search is an attractive approach because we can reprioritize the order in which features are selected for updating according to their correlation with the error *and* their associated feature cost. Even though the parsimonious linear regression optimization problem is quite different from the lasso objective, algorithmically, only minor modifications are required to approximate the solution to parsimonious linear regression. In fact, all that is needed is the adjustment of the score used to select the next appropriate feature with respect to the costs of the features (See step 6(a) of the algorithm below). We present a modified least angle regression algorithm called ParLiR to approximate the solution of parsimonious linear regression in Figure 1.

However, it is not immediately clear that modified least angle regression techniques will still produce a low-cost solution to the parsimonious linear regression problem. Therefore, we experimentally evaluate efficiency, approximation error and parsimony in Section 4 and prove formal theoretical guarantees on parsimony and a special case of optimality in Section 5.

```
Parsimonious Linear Regression Approximation (ParLiR)

1. Input: a set $\mathcal{D}$ of $m$ data samples represented as
   $n$ $m$-length feature vectors $\vec{f}_1, \dots, \vec{f}_n$ and an $m$-length target vector $\vec{y}$.

2. Initialize the step-size $\eta$ to some small positive value.

3. Initialize the current selected feature set $\mathcal{F} = \emptyset$.

4. Initialize weight vector $\vec{w} = \langle w_0, w_1, \dots, w_n \rangle$
   with $w_0$ equal to the average value of $\vec{y}$ (this will
   give the residuals a mean of 0), and $w_{i>0} = 0$.

5. Define the residual vector $\vec{r}_{\vec{w}}$ for the current
   weight settings as the following:

   $$\vec{r}_{\vec{w}} = \vec{y} - \left[ w_0 + \sum_{i=1}^{n} w_i \vec{f}_i \right]$$

6. Repeat the following:

   (a) Calculate the cost-penalized correlation
       score for all $f_i \in \mathcal{F}$:

       $$score_i = \frac{1}{m} \left| \vec{f}_i \cdot \vec{r}_{\vec{w}} \right| - \mathbb{I}[f_i \in \mathcal{F}]\sqrt{c_i}$$

   (b) Find the feature $f_i$ with the highest
       $score_i \geq \eta$; if no such feature found then
       halt and Output: $\vec{w}$.

   (c) If $f_i \notin \mathcal{F}$, let $\mathcal{F} = \mathcal{F} \cup \{f_i\}$ and let $w_i =$
       $w_i + \mathrm{sgn}(\vec{f}_i \cdot \vec{r}_{\vec{w}})\sqrt{c_i}$. (see footnote[2])

   (d) Else let $w_i = w_i + \mathrm{sgn}(\vec{f}_i \cdot \vec{r}_{\vec{w}})\eta$.
```

Figure 1. The ParLiR Algorithm

3.3 Time complexity of ParLiR

Finding the optimal cost-sensitive linear function for the given data requires examining every subset of features, as the possible inclusion of a feature has a non-monotonic, non-convex influence on the total score. This implies that finding the exact solution is NP-complete, and a straightforward algorithm to find it (considering every subset and calculating its score on the training data) has time complexity $\mathcal{O}(m\#F2^{\#F})$. If we look at the time complexity of ParLiR, it can be seen that all data has to be checked for every update. Each such a check takes time of the order of $\mathcal{O}(m\#F)$. This is repeated for every selected feature and every weight update. The number of weight updates for a given feature is of the order of $\mathcal{O}(\frac{1}{\eta})$, and the number of selected features is equal to $\#\mathcal{F}$. The total time complexity of ParLiR is then $\mathcal{O}(\frac{\#\mathcal{F}m\#F}{\eta})$.

[2] $\mathrm{sgn}(\cdot)$ is $+1$ if its argument is non-negative and -1 otherwise.

3.4 Dealing with Missing Values

In the proposed setting, it is important to differentiate between two possible versions of missing values. Since features are costly, it is possible the feature is missing because it wasn't requested. Because the decision not to pay for a feature when collecting training data should not influence its correlation with the target value, we do not count examples of this first case when computing the score of a feature as described in step 6(a) in the algorithm above. I.e., m is set to the number of training examples minus the number of examples where we decided not to pay for the feature.

However, it is also possible the feature was requested and payed for but the associated test failed and we didn't receive the feature value, for example when an experiment failed. In this case, we set the feature value to 0 (as stated, we assume all features to have a 0 mean) and do count the example when computing the correlation. If a feature does not give any result with a probability p, the correlation will be multiplied by a factor $(1-p)$. This can be regarded as an increase of the cost of the feature by a factor $\frac{1}{1-p}$, i.e. a feature likely to fail is, in a way, more costly.

4 Experiments

We performed experiments on several different data-sets from the UCI repository (also available on Weka [9]). We report the results obtained for the 'Pima-Indians diabetes', 'Boston Housing' and 'Bodyfat' data-sets. Results for the other datasets were comparable but less illustrative. The first of these is a classification data-set, the other are regression tasks. For the regression data sets, there are no feature costs given but we decided to to use artificial costs based on our own intuition, i.e., medical experiments are more costly than information such as *age* and *gender* for example.

To show the behavior of the different algorithms for varying relative costs, we multiplied the basic cost vectors by a varying factor.

All reported results in this section are averages of 10 repetitions of 10-fold cross-validation using a random partitioning of the data for each repetition.

The plots given in this section all show the $C(\vec{w}) + E(\vec{w}, \mathcal{D})$ measure for varying cost-factors for both the training data and test-data. We compare our algorithm with another cost-sensitive linear regression algorithm, which is based on greedy forward-selection. Here the set of features is found by taking the highest scoring feature, projecting the target vector to the hyperplane orthogonal to the feature vector, and repeating until no feature has a positive score. The subset of features selected in this way is then used in normal linear least-squares regression.

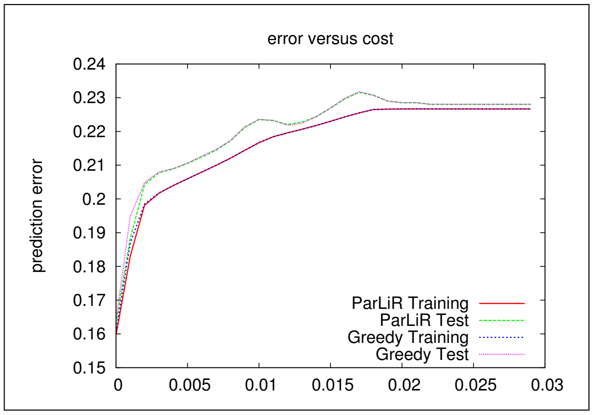

Figure 2. The Pima-Indians dataset.

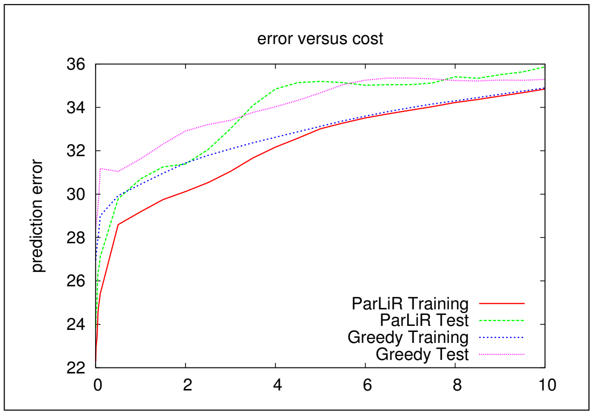

Figure 3. The Housing dataset.

4.1 Pima-indians Diabetes dataset

This dataset contains user-defined costs (reflecting the actual price of medical tests according to the Ontario Health Insurance Program). The dataset has eight numerical features. The target is either 0 or 1, indicating whether the patient tested negative or positive for diabetes.[3] Most of the costs are equal to 1.0, except for two tests (glucose test and insulin test). The results of our algorithm can be seen in Figure 2. It is clear that both algorithms perform almost equally well, both on training data and test-data, with a slight advantage for ParLiR on the training data. The behavior is characteristic: for low costs, all features are used. As costs increase, only a few (cheap) features are used. From a given point (cost-factor $= 0.02$) the algorithm prefers the prediction error cost to the cost of the features.

[3] We note that this dataset was meant for classification. As we had no regression dataset with user-defined costs, the result of normal regression is given as a proof of concept. For a classification task it is advisable not to use basic linear regression, but instead to use logistic regression.

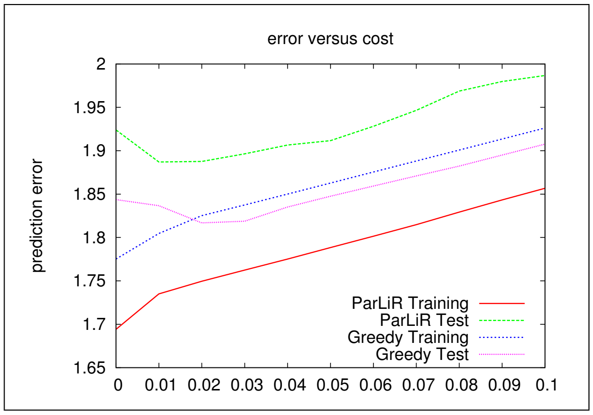

Figure 4. The Bodyfat dataset.

4.2 Housing dataset

The **housing** dataset contains information about different suburbs of Boston, such as crime rate, distance to employment centers, number of rooms per dwelling, . . . The target function is the average housing value. For this set we defined costs according to the confidentiality of the information: the amount of tax paid is more costly than the teacher-to-pupil ratio, for example.

The results can be seen in Figure 3. For this set it is more clear that ParLiR performs better on the training data. The behavior for the test data shows that the either algorithm might be better for specific values of the cost factor. The critical point where all information is too expensive is not reached in this figure.

4.3 Bodyfat dataset

The **bodyfat** dataset is used for predicting the percentage of bodyfat using different measures of the body. We gave the attribute 'age' a cost equal to 0, the measuring of density from underwater weighing a cost of 1.0 (indicating the more elaborate procedure needed) and all other attributes a cost of 0.5. From Figure 4 we can see that for this dataset there is a large difference between the two algorithms. ParLiR clearly outperforms the greedy selection algorithm on the training data, however on the test data it is the other way round. This might imply that the dataset gives rise to overfitting when linear regression is used.

4.4 Summary of experiments

From all experiments it was clear that ParLiR performs very well on the training data. The proof of parsimonity as given in section 5 explains this: for the training data, every feature used by ParLiR will pay back at least its cost in the training set. For the greedy selection algorithm no such property holds.

The greedy selection algorithm outperformed ParLiR on the test data in some experiments, however. For most of the datasets, this might be due to them not being well-suited for linear regression, leading to bad fitting. Comparing to the optimal solution (based on the training data) was not feasible as such an algorithm takes time exponential in the number of features.

5 Theoretical Results

5.1 Proof of Parsimony

Theorem 5.1 (Parsimony of ParLiR)**.** *Every feature which is introduced in step 6c of the ParLiR algorithm immediately reduces the mean squared error of the prediction by the value of its cost. Furthermore, at every weight update in step 6d, the mean squared error is reduced by* η^2.

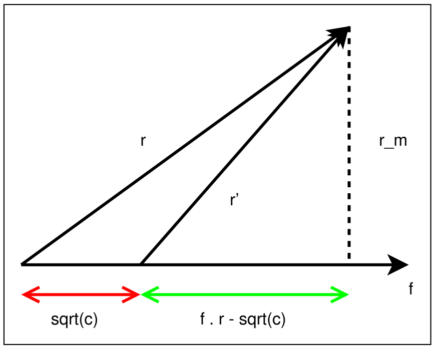

Figure 5. Geometric representation for clarification of the proof

Proof. For the proof we refer to figure 5.1 for clarification. Important to note is that the correlation between two vectors, $\vec{f_i} \cdot \vec{r}$ (when $\|\vec{f_i}\|_2 = 1$) is equal to the length of the orthogonal projection of $\vec{r}$ on $\vec{f_i}$. In the figure we use $\vec{r}, \vec{r'}$ and $\vec{r_m}$ to respectively indicate the current residual at the moment the feature f_i is selected, the residual after w_i has been updated by $\sqrt{c_i}$ and the minimal residual we can obtain by only changing the weight of feature f_i, which we get for w_i equal to the correlation: the minimal distance from the target to the feature vector is equal to the orthogonal distance. We give the proof for $\vec{f_i} \cdot \vec{r} > 0$, the case where $\vec{f_i} \cdot \vec{r} < 0$ is completely analogous up to some changes in sign.

We divide the residuals by the number of samples in the batch, to indicate how much the error *per sample* relates to the cost spent on each sample.

In the theorem we will use the *average* correlation of all data samples, $\frac{\vec{f_i} \cdot \vec{r}}{m}$, instead of the normal correlation, as this gives an indication how the average error over all samples will decrease.

We will first prove the statement about step 6c. Note that, as the feature is added, step 6b of the algorithm tells us that $\frac{|\vec{f_i} \cdot \vec{r}|}{m} > \sqrt{c_i}$ must hold.

Using the theorem of Pythagoras, we get:

$$\begin{aligned} \|\frac{\vec{r}}{m}\|_2^2 &= \|\frac{r_m}{m}\|_2^2 + (\frac{\vec{f_i} \cdot \vec{r}}{m})^2 \\ \|\frac{\vec{r'}}{m}\|_2^2 &= \|\frac{r_m}{m}\|_2^2 + (\frac{\vec{f_i} \cdot \vec{r}}{m} - \sqrt{c_i})^2 \end{aligned}$$

Rewriting this gives:

$$\begin{aligned} & |\frac{\vec{r}}{m}\|_2^2 - \|\frac{\vec{r'}}{m}\|_2^2 \\ & = (\frac{\vec{f_i} \cdot \vec{r}}{m})^2 - (\frac{\vec{f_i} \cdot \vec{r}}{m})^2 + 2\sqrt{c_i}\frac{\vec{f_i} \cdot \vec{r}}{m} - c_i \\ & > 2\sqrt{c_i}\sqrt{c_i} - c_i \text{ (because } \frac{\vec{f_i} \cdot \vec{r}}{m} > \sqrt{c_i}) \\ & = c_i \end{aligned}$$

For the case where step 6d is performed, the proof is analogous: we only need to substitute η for $\sqrt{c_i}$. □

5.2 Approximation Error Bound in the Case of Orthogonal Features

In this part we consider the case where all features are mutually independent. We prove that in this case, the ParLiR algorithm will closely approximate the exact solution.

Theorem 5.2 (Error bound on ParLiR in the case where features are mutually orthogonal)**.** *In the case where feature vectors are mutually orthogonal, i.e.* $\vec{f_i} \cdot \vec{f_j} = 0$ *if* $i \neq j$*, the ParLiR algorithm gives an approximation closer than* $k\eta^2$ *to the optimal solution to the parsimonious linear regression problem, with* k *the number of features which has a non-zero weight in the optimal solution.*

The proof is intuitive and based on the fact that in the case of orthogonal features, updating the weight of one feature does not influence the score of another. This implies that all included features have weights closer than η to the optimal weights. Applying the generalized theorem of Pythagoras gives the required result.

6 Conclusions and Future Work

In this paper we formalize the problem of linear regression where features are only observable at a certain cost. We argued that finding the exact solution for this parsimonious linear regression problem is computationally hard for large datasets using existing mixed integer quadratic programming approaches. We introduced ParLiR, an adaptation of a least-angle regression algorithm, which efficiently finds an approximation to the solution. We have shown both empirically and theoretically that the solution found by this algorithm is cost-efficient.

There are various routes for future work. As became clear from the empirical results, linear regression is not well-suited for many domains, and more complex regression functions might be more appropriate. We have the feeling that many regression methods might be adapted to be cost-sensitive, such as artificial neural networks, support vector machines and kernel methods.

In the current setup, we use the same costly features for each example. It might be useful to employ the cost-less features to determine which costly features to use, thereby making the selection of features different for each sample. This possibly allows for a better resolution. Having a complex regression function which is adaptable according to the cost-free information would give a powerful, cost-efficient method for parsimonious linear regression.

References

[1] P. Brown, T. Fearn, and M. Vannucci. The choice of variables in multivariate regression: A non-conjugate bayesian decision theory approach. *Biometrika*, 86:635–648, 1999.

[2] P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining*, pages 155–164, 1999.

[3] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *ANNALS OF STATISTICS*, 32:407, 2004.

[4] C. Elkan. The foundations of cost-sensitive learning. In *IJCAI*, pages 973–978, 2001.

[5] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.

[6] L. Torgo and R. Ribeiro. Utility-based regression. In *Principles of Knowledge Discovery and Data Mining*, pages 597–604, 2007.

[7] P. D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, March 1995.

[8] P. D. Turney. Types of cost in inductive concept learning. In *Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning (ICML-2000)*, Stanford, California, 2000.

[9] Weka 3: Data mining software in java, 2004. The University of Waikato, Dept. of Computer Science, Machine Learning lab. http://www.cs.waikato.ac.nz/ml/weka/.