REGRESSION BY SELECTING APPROPRIATE FEATURE(S)

Tolga Aydın and H.Altay Güvenir

Department of Computer Engineering
Bilkent University
Ankara, 06533, TURKEY

Abstract. This paper describes two machine learning methods, called *Regression by Selecting Best Feature Projection* (RSBFP) and *Regression by Selecting Best Features* (RSBF). RSBFP projects the training data on each feature dimension and produces exactly one simple linear regression line on each continuous feature. In the case of categorical features, exactly one simple linear regression line per each distinct value of each categorical feature is produced. Then these simple linear regression lines are analyzed to determine the feature projection that is best among all projections. The best feature projection and its corresponding simple linear regression line is then selected to make predictions. RSBF consists of two phases: The first phase uses RSBFP to sort predictive power of each feature projection. The second phase calls multiple linear least squares regression for number of features times, each time excluding the worst feature among the current set, and produces multiple linear regression lines. Finally, these regression lines are analyzed to select the best subset of features. The best subset of features and its corresponding multiple linear regression line is then selected to make predictions.

Keywords: Prediction, Feature Projection, Regression.

1 INTRODUCTION

Prediction has been one of the most common problems researched in data mining and machine learning. Predicting the values of categorical features is known as classification, whereas predicting the values of continuous features is known as regression. From this point of view, classification can be considered as a subcategory of regression. In machine learning, much research has been performed for classification. But, recently the focus of researchers has moved towards regression, since many of the real-life problems can be modeled as regression problems.

There are two different approaches for regression in machine learning community: Eager and lazy learning. Eager regression methods construct models by using the training data, and the prediction task is based on these models. The advantage of eager regression methods is not only the ability to obtain the interpretation of the underlying data, but also the reduced test time. On the other hand, the main disadvantage is their long train time requirement. Lazy regression methods do not construct models by using the training data. Instead, they delay all processing to prediction phase. The most important disadvantage of lazy regression methods is the fact that, they do not enable interpretation of the training data. Because the model is usually the training data itself. It is not a compact description of the training data, when compared to the models produced by eager regression methods, such as regression trees and rule based regression.

In the literature, many eager and lazy regression methods exist. Among eager regression methods, CART [1], RETIS [7], M5 [5], DART [2], and Stacked Regressions

[9] induce regression trees, FORS [6] uses inductive logic programming for regression, RULE [3] induces regression rules, and MARS [8] produces mathematical models. Among lazy regression methods, *k*NN [4, 10] is the most popular nonparametric instance-based approach, RFP [11] and RPFP [14] are nonparametric approaches based on feature projections.

In this paper, we describe two eager learning methods, namely *Regression by Selecting Best Feature Projection* (RSBFP) and *Regression by Selecting Best Features* (RSBF). Both of the methods make use of the linear least squares regression.

RSBFP produces projections of the train data on each feature. For each continuous feature, a simple linear regression line is produced using simple linear least squares regression. Since it is impossible to employ simple linear least squares regression on categorical features, a different approach is used for them. A simple least squares regression is performed for each distinct value of each categorical feature to produce simple linear regression lines. At the end of the train phase, these simple linear regression lines, from both type of features, are sorted according to their prediction ability. In the test phase, the target value of a test instance is predicted using the simple linear regression line having the minimum relative error, i.e. having the maximum predictive power.

RSBF consists of two phases. In the first phase, RSBFP is employed to sort the predictive power of each feature. In the second phase, multiple linear least squares regression is employed for number of features times, each time excluding the worst feature among the current set. The test phase is similar to the RSBFP's test phase. That is, the target value of a test instance is predicted using the multiple linear regression line having the minimum relative error, i.e. having the maximum predictive power.

Although the domain of the applicability of RSBFP is not restricted, RSBF suffers at this point. Current version of RSBF does not handle missing feature values and categorical features. The reason for not handling categorical features is the fact that multiple linear least squares regression can be employed only on continuous features. For any training instance to be used in a multiple linear least squares regression, all of its feature values must be known. This training instance becomes inappropriate for multiple linear least squares regression even in the existence of just one missing feature value. Therefore, to avoid loss of information, RSBF is employed on domains including no missing feature values.

In this paper, RSBF and RSBFP are compared with 3 eager (RULE, MARS, DART) and 3 lazy methods (RFP, RPFP, *k*NN) in terms of predictive power and computational complexity. The predictive power of both methods are comparable to other eager methods. But the most important result is the fact that both methods are the fastest among all eager and lazy methods. For most data mining or knowledge discovery applications, where very large databases are in concern, this is thought of a solution because of low computational complexity.

In Section 2, we review the *k*NN, RFP, RPFP, RULE, MARS and DART methods for regression. Section 3 and Section 4 give a detailed description of the RSBF and RSBFP, respectively. Section 5 is devoted to the empirical evaluation of RSBF, RSBFP and their comparisons with other methods. Finally, in Section 6, conclusions and future works are presented.

2 REGRESSION OVERVIEW

*k*NN is the most commonly used lazy method for both classification and regression problems. The underlying idea behind the *k*NN method is that the closest instances to the query point have similar target values with the query. Hence, the *k*NN method first finds the closest instances to the query point in the instance space according to a distance measure. Generally, the Euclidean distance metric is used to measure the similarity between two points in the instance space. Therefore, by using Euclidean distance metric as our distance measure, k closest instances to the query point are found. Then kNN outputs the weighted average of the target values of those closest instances as the prediction for that query instance.

Regression by Feature Projections (RFP) is another lazy method, where the instances are stored as their projections on each feature dimension. In this method, the kNN method is used on each individual feature dimension to find their own prediction, independent of the predictions of other features. The final prediction is made by combining individual feature predictions. Here, the contribution of each feature changes according to the local position of the query instance. The prediction time requirement of RFP is much lower than kNN.

Regression by Partitioning Feature Projections (RPFP) gives a different approximation for each feature by using the projection of instance space to each feature dimension separately. These approximations may be different for each feature for a single query point. Up to this point, RPFP looks like RFP. But the main difference lies in the partitioning process employed by RPFP. A partitioning strategy is employed in the method and some portion of the data is removed from the instance space. The same approximations are repeated for a sequence of partitioning steps, where the partitioning continues until reaching a small number of instances and meanwhile records of previous steps are kept.

In machine learning, inducing rules from a given train data is also popular. Weiss and Indurkhya adapted the rule-based classification algorithm [12], Swap-1, for regression. Swap-1 learns decision rules in Disjunctive Normal Form (DNF). Since Swap-1 is designed for the prediction of categorical features, using a preprocessing procedure, the numeric feature in regression to be predicted is transformed to a nominal one. For this transformation, the P-class algorithm is used [3]. If we let $\{y\}$ a set of output values, this transformation can be regarded as a one-dimensional clustering of training instances on response variable y, in order to form classes. The purpose is to make y values within one class similar, and across classes dissimilar. The assignment of these values to classes is done in such a way that the distance between each y_i and its class mean must be minimum. After formation of pseudo-classes and the application of Swap-1, a pruning and optimization procedure can be applied to produce an optimum set of regression rules.

MARS method partitions the training set into regions by splitting the features recursively into two regions, by constructing a binary regression tree. MARS is continuous at the borders of the partitioned regions. It is an eager, partitioning, interpretable and an adaptive method.

DART, also an eager method, is the latest regression tree induction program developed by Friedman [13]. It avoids limitations of disjoint partitioning, used for other tree-based regression methods, by producing overlapping regions with increased training cost.

3 REGRESSION BY SELECTING BEST FEATURE PROJECTION (RSBFP)

RSBFP method tries to determine the feature projection that achieves the highest prediction accuracy. The next subsection describes the training phase for RSBFP, then we describe the testing phase.

3.1 Training

Training in RSBFP begins simply by storing the training data set as projections to the features. A copy of the target values is associated with each projection and the training data set is sorted for each feature dimension according to their feature values. If a training instance includes missing values, it is not simply ignored. Instead, that training instance is stored for the features on which its value is given. The next step involves producing the simple linear regression lines for each feature. This step differs for categorical and continuous features. In the case of continuous features, exactly one simple linear regression line per continuous feature is produced. On the other hand, the number of simple linear regression lines per each categorical feature is the number of distinct feature values at the feature of concern. For categorical features, the parametric form of the simple regression line is constant. This issues can be illustrated through an example.

Let our example domain consist of four features, f_1, f_2, f_3 and f_4 , where f_1, f_2 are continuous and f_3, f_4 are categorical. For continuous features, we define *minvalue* [f] and *maxvalue* [f] to denote the minimum and maximum value of feature f, respectively. Furthermore, *No_categories* [f] is defined to give the number of distinct categories of feature f, for categorical features. In our example domain, let the following values be observed:

minvalue [f_1] = 4, *maxvalue* [f_1] = 10
minvalue [f_2] = 2, *maxvalue* [f_2] = 8
No_categories [f_3] = 2 (A, B)
No_categories [f_4] = 3 (X, Y, Z)

For this example domain, 7 simple linear regression lines are produced: 1 for f_1, 1 for f_2, 2 for f_3, and finally 3 for f_4. Let the following be the parametric form of the simple linear regression lines:

Simple linear regression line for f_1:	target = $2f_1 - 5$
Simple linear regression line for f_2:	target = $-4f_2 + 7$
Simple linear regression line for A category of f_3:	target = 6
Simple linear regression line for B category of f_3:	target = -5
Simple linear regression line for X category of f_4:	target = 10
Simple linear regression line for Y category of f_4:	target = 1
Simple linear regression line for Z category of f_4:	target = 12

The training phase is completed by sorting these simple linear regression lines according to their predictive power. The relative error (RE) of the regression lines is used as the indicator of predictive power: the smaller the RE, the stronger the predictive power. RE of each simple linear regression line is computed by the following formula:

$$RE = \frac{MAD}{\frac{1}{Q}\sum_{i=1}^{Q}|t(q_i) - \bar{t}|}$$

where Q is the number of training instances used to produce the simple linear regression line, $\bar{t}$ is the median of the target values of Q training instances, $t(q_i)$ is the actual target value the ith training instance The MAD (Mean Absolute Distance) is defined as follows:

$$MAD = \frac{1}{Q}\sum_{i=1}^{Q}|t(q_i) - \hat{t}(q_i)|$$

Here, $\hat{t}$ (q_i) denotes the predicted target value of the ith training instance according to the induced simple linear regression line.

We had 7 simple linear regression lines, and let's suppose that they are sorted as the following, from the best predictive to the worst one: f_3-A, f_4-X, f_2, f_1, f_4-Y, f_4-Z, and finally, f_3-B. This shows that any categorical feature's predictive power varies among its categories. For the above sorting schema, categorical feature f_3 is both the best and the worst feature. Its predictions are reliable among its category A, although it is very poor among category B.

3.2 Testing

In order to predict the target value of a test instance t_i , the RSBFP method uses exactly one simple linear regression line. This line may not always be the best one. The reason for this situation is explained via an example. Let the feature values of the test instance t_i be as the following:

$f_1(t_i) = 5$, $f_2(t_i) = 10$, $f_3(t_i) = B$, $f_4(t_i) =$ missing

Although the best simple linear regression line is f_3-A, this line can not be used for our t_i , since $f_3(t_i) \neq$ A. The next best simple linear regression line, which is worse than only f_3-A, is f_4-X. This line is inappropriate for our t_i , as well. No prediction can be made for missing feature values. ($f_4(t_i)$ = missing). Therefore, the search for the best simple linear regression line, continues. The line produced by f_2 comes next. It is again not possible to benefit from this simple linear regression line. Because $f_2(t_i) = 10$, and it is not in the range of f_2, (2, 8). Fortunately, we find an appropriate regression line in the 4th trial. Our $f_1(t_i)$, which is 5, is in the range of f_1, (4, 10). This simple linear regression line computes target as $2f_1 + 5$. So the prediction made for target value of t_i is $(2 * f_1(t_i) + 5) = (2 * 5 + 5) = 15$. Once the appropriate regression line is found, remaining regression lines need not be dealed anymore.

4 REGRESSION BY SELECTING BEST FEATURES (RSBF)

RSBF method tries to determine a subset of the features such that this subset consists of the best features. Its applicability is restricted for the reasons mentioned in Section 1. The next subsection describes the training phase for RSBF, then we describe the testing phase.

4.1 Training

The training phase of RSBF consists of two phases. The first phase is exactly the same as the training phase of RSBFP with the exception that, the data is assumed to be free from missing values and categorical features. At the end of this first phase of training, continuous features are sorted according to their predictive power. In the second phase of training, multiple linear least squares regression is employed for number of features times, each time excluding the next worst feature of the current set. It will be suitable to describe the second phase through an example.

Let our example domain consist of three continuous features, f_1, f_2, f_3 and assume that they are sorted as f_2, f_3, f_1 according to their predictive power at the end of the first phase of training. We begin employing multiple linear least squares regression on all 3 features. The output of this process is a multiple linear regression line involving contributions of all three features. This line is denoted by $MLRL_{1,2,3}$. Then we exclude the worst feature, namely f_1, and run multiple linear least squares regression to obtain $MLRL_{2,3}$. In the final step, we exclude the next worst feature of the current set, namely f_3, and obtain $MLRL_2$. Actually the multiple linear least squares regression transforms into simple linear least squares regression in the final step, since we deal with exactly one feature.

The second phase of training is completed by sorting MLRLs according to their predictive power, the smaller the RE of a MLRL, the stronger the predictive power of that

MLRL. The computation of RE of a MLRL is exactly the same as that of a simple linear regression line, mentioned in Section 3.

4.2 Testing

In order to predict the target value of a test instance t_i , the RSBF method uses exactly one multiple linear regression line (MLRL). This line is always the best one. In the example, given in Section 4.1, let's suppose that the parametric form of $MLRL_{2,3}$ is $(4f_2 - f_3 + 3)$ and that MLRLs are sorted as the following: $MLRL_{2,3}$, $MLRL_2$ and $MLRL_{1,2,3}$. RSBF method will choose $MLRL_{2,3}$ and its parametric form $(4f_2 - f_3 + 3)$ to determine the target values of all test instances. In contrast to RSBFP, RSBF does not require the feature values of test instances to be in a range to accomplish the prediction.

5 EMPIRICAL EVALUATION

RSBFP and RSBF methods were compared with the other methods in terms of predictive accuracy and time complexity. Predictive accuracy is measured in terms of relative error (RE). We have used two different data sets in our experiments: one consisting of 27 data files, and the other consisting of 9 data files. In fact, the latter set is a subset of the former. The data files in the second set include no missing values and no categorical features. So this set is appropriate for RSBF to work with. The characteristics of the data files are summarized in Table1.

10 fold cross-validation technique was employed in the experiments. For lazy regression methods *k* parameter was taken as 10, where *k* denotes the number of nearest neighbors considered around the query instance. Table 2 shows the relative errors of the 7 methods on the first set. RSBFP is the best in 5 data files. This is an improvement since no other method is the best in more than 5 data files. The characteristics of these 5 data files are as the following: 4 of them include missing values, 3 of them include categorical features, 2 of them include both. The number of data files including missing values, categorical features and both is 12, 12 and 6, respectively. From this information, it is seen that 4/12 (%33) of the data files including missing values, 3/12 (%25) of the data files including categorical features and 2/6 (%33) of the data files including both can best be predicted by RSBFP. But the predictive power of RSBFP is noted to be better than only RULE and MARS, if we are concerned with average RE.

Table 5 shows the relative errors of the 8 methods on the second set. RSBF is the best in 2 data files. This is also an improvement since only RPFP method is the best in more than 2 data files. If we are concerned with average RE, the predictive power of RSBF is noted to be better than MARS, RFP, RULE and RSBFP. The predictive power of RSBFP decreases, since it is now better than only MARS . These results indicate that RSBFP is not appropriate for domains including neither missing values nor categorical features. But in the presence such domains, RSBF may be a good choice, since it gives promising results.

In machine learning, an algorithm's success is not only determined by its predictive power, but also by its time complexity. Therefore, the real importance of RSBFP and RSBF methods lies in their low execution time requirements. Table 3 and 6 show the training time durations, whereas Table 4 and 7 show the test time durations of the proposed and existing methods. Both of the proposed methods, RSBFP and RSBF, have less training time requirements than other eager methods, MARS, DART and RULE. It is not possible to be faster than lazy methods RFP, RPFP and *k*NN in the train phase, because lazy regression methods delay all processing to the test phase.

In the test phase, both RSBFP and RSBF are better than the lazy methods. They are better than RULE, an eager regression method, too. The total running time involves both training and test time. In that aspect, RSBFP becomes the first, *k*NN the second and RSBF the third fastest method. Many real-world data sets consist of huge number of instances that need to be processed efficiently. Therefore, data mining and machine learning community always searches for as efficient as possible approaches. The two proposed approaches mentioned in this paper handle this problem, although they aren't perfect in prediction process.

6 CONCLUSIONS AND FUTURE WORK

We have described two eager regression methods, Regression by Selecting Best Feature Projection (RSBFP) and Regression by Selecting Best Features (RSBF), which achieve fast computation time, by preserving a comparable or better accuracy with other popular eager and lazy regression methods. They also enable the interpretation of the data, which is desired by data mining community.

RSBF selects the best features to form a parametric model, namely multiple linear regression line. On the other hand, RSBFP determines a unique feature projection that is best among all projections and makes use of it.

RSBFP is suitable for domains including missing feature values and categorical features. On the other hand, RSBF is strong on domains including neither missing values nor categorical features. From this perspective, these two methods seem to complement each other. The RSBF method can be modified to handle missing values and categorical features as a future work.

Table1. Characteristics of the data files used in the empirical evaluations. C: Continuous, N: Nominal

Dataset	**Original Name**	**Instances**	**Features (C+N)**	**Missing Values**
AB	Abalone	4177	8 (7 + 1)	None
AI	Airport	135	4 (4 + 0)	None
AU	Auto-mpg	398	7 (6 + 1)	6
BA	Baseball	337	16 (16 + 0)	None
BU	Buying	100	39 (39 + 0)	27
CL	College	236	25 (25 + 0)	381
CO	Country	122	20 (20 + 0)	34
CP	Cpu	209	7(1 + 6)	None
EL	Electric	240	12 (10 + 2)	58
FA	Fat	252	17 (17 + 0)	None
FI	Fishcatch	158	7 (6 + 1)	87
FL	Flare2	1066	10 (0 + 10)	None
FR	Fruitfly	125	4 (3 + 1)	None
GS	Gss2	1500	43 (43 + 0)	2918
HO	Home Run Race	163	19 (19 + 0)	None
HU	Housing	506	13 (12 + 1)	None
NO	Normal Temp.	130	2 (2 + 0)	None
NR	Northridge	2929	10 (10 + 0)	None
PL	Plastic	1650	2 (2 + 0)	None
PO	Poverty	97	6 (5 + 1)	6
RE	Read	681	25 (24 + 1)	1097
SC	Schools	62	19 (19 + 0)	1
SE	Servo	167	4 (0 + 4)	None
ST	Stock Prices	950	9 (9 + 0)	None
TE	Televisions	40	4 (4 + 0)	None
UN	Usnews Coll.	1269	31 (31 + 0)	7624
VL	Villages	766	32 (29 + 3)	3986

Table2. Relative Errors of Algorithms. Best results are typed with bold font

Dataset	RSBFP	RFP	RPFP	KNN	RULE	MARS	DART
AB	0.788	0.748	0.675	**0.661**	0.899	0.683	0.678
AI	0.593	0.499	**0.473**	0.612	0.744	0.720	0.546
AU	0.510	0.426	0.334	**0.321**	0.451	0.333	0.346
BA	0.720	0.787	0.653	**0.443**	0.666	0.493	0.508
BU	**0.656**	0.911	0.840	0.961	0.946	0.947	0.896
CL	0.807	1.001	0.692	0.764	0.290	1.854	**0.252**
CO	2.524	1.439	**1.301**	1.642	6.307	5.110	1.695
CP	0.882	0.766	0.625	0.944	0.678	0.571	**0.510**
EL	**1.008**	1.032	1.009	1.194	1.528	1.066	1.118
FA	0.720	0.887	0.667	0.785	0.820	**0.305**	0.638
FI	0.599	0.540	0.352	0.697	0.355	**0.214**	0.415
FL	2.324	1.368	**1.218**	2.307	1.792	1.556	1.695
FR	**1.010**	1.065	1.056	1.201	1.558	1.012	1.077
GS	0.672	0.768	0.566	0.654	**0.218**	0.359	0.410
HO	0.883	1.000	0.868	0.907	0.890	**0.769**	0.986
HU	0.887	0.798	0.618	0.600	0.641	0.526	**0.522**
NO	0.969	1.040	**0.962**	1.232	1.250	1.012	1.112
NR	1.046	0.984	0.947	1.034	1.217	0.928	**0.873**
PL	0.833	0.895	0.415	0.475	0.477	**0.404**	0.432
PO	0.931	**0.670**	0.703	0.796	0.916	1.251	0.691
RE	**1.004**	1.011	1.008	1.062	1.352	1.045	1.189
SC	0.310	0.404	0.319	0.388	0.341	**0.223**	0.352
SE	0.710	0.822	0.527	0.619	**0.229**	0.432	0.337
ST	1.731	0.992	0.729	**0.599**	0.906	0.781	0.754
TE	4.577	4.014	**1.659**	1.895	4.195	7.203	2.690
UN	**0.395**	0.714	0.666	0.480	0.550	0.412	0.623
VL	1.017	**0.967**	0.970	1.017	1.267	1.138	1.355
Mean	1.078	0.983	**0.772**	0.900	1.166	1.161	0.841

Table 3. Train time of algorithms in milliseconds. Best results are typed with bold font

Dataset	RSBFP	RFP	RPFP	KNN	RULE	MARS	DART
AB	201	144.2	174.7	**8.9**	3219	10270	477775
AI	2.2	1	1.2	**0**	90.8	159.2	62
AU	11.7	7.7	9.5	**0.6**	248.9	570.5	1890.1
BA	24.4	14.9	21	**0**	181.8	915.1	3171.1
BU	15.4	9.5	15.1	**0**	67.1	761.7	794.4
CL	23.6	14.4	21.4	**0.5**	148.2	1274.3	717.6
CO	9.4	5.4	8.6	**0.1**	108.6	475.3	481
CP	5.5	3.6	4.1	**0**	52.7	575.3	286
EL	11.5	7.1	9.4	**0.2**	69.5	407.5	1017
FA	19	11.4	16	**0**	161.1	985	1773.9
FI	3	2.1	3.1	**0**	47.8	240.2	201.4
FL	45.3	30.6	39.7	**3.5**	108.8	667.2	971.4
FR	1.1	0.7	1.1	**0**	34.1	99.5	45.9
GS	337.5	217.4	303.9	**13.5**	862.8	10143.9	27266
HO	11.6	6.5	10	**0**	57.5	616.3	893.9
HU	28.7	18	26.4	**1**	264.9	1413.9	8119.7
NO	**0**	0.1	**0**	**0**	30.6	69.3	18.9
NR	173.7	98	128.2	**7.4**	3493	5709.9	87815
PL	18.4	12.7	15.7	**0.2**	175.3	824.8	10024.4
PO	2.1	1	1.7	**0**	40.9	127.3	44
RE	73.4	47.2	67.5	**3**	196	2744.6	33044.6
SC	4	2.4	4.2	**0**	45.3	260.8	84.4
SE	2.3	1	2.2	**0**	37	116.4	83.4
ST	41.3	26.6	33.4	**1.4**	365.1	2281.4	17346.4
TE	**0**	**0**	**0**	**0**	30.9	31.1	3.1
UN	181.8	114.8	186	**7.4**	2547.1	8435.2	168169
VL	94.3	57.8	95.6	**4.4**	513.6	3597.8	23405
Mean	49.711	31.707	44.433	**1.9296**	488.83	1991.61	32055.7

Table 4. Test time of algorithms in milliseconds. Best results are typed with bold font

Dataset	RSBFP	RFP	RPFP	KNN	RULE	MARS	DART
AB	27.8	201.6	43994	6547	14433.1	7.9	**6.1**
AI	1	3.2	8.9	3.4	141.7	**0**	**0**
AU	2.4	14.6	139.5	64.5	462.2	**0**	**0**
BA	2.3	29.4	298.3	54.6	244.8	**0**	**0**
BU	0	11.9	69.5	11.6	32.1	**0**	**0**
CL	1.1	22.9	168.5	38.2	40.3	1	**0**
CO	0.4	12.3	35.4	8.4	98.4	**0**	0.1
CP	1	6.4	31.7	11.6	87.3	**0**	**0**
EL	1.5	13.7	143.2	21	117.5	**0**	**0**
FA	1.3	23.7	187.6	33.1	96.4	**0**	**0**
FI	1	5.5	18.9	7.9	48.8	**0**	**0**
FL	4.2	86.8	1387	407.8	223.6	0.4	**0**
FR	0.2	2.5	10.2	2	45.4	**0**	**0**
GS	10.2	517.1	16008.5	2699.7	312.3	2.7	**1.7**
HO	0.4	10.9	104	13.3	43	**0**	**0**
HU	3.2	36.9	503.2	107.8	410.5	**0**	**0**
NO	0.2	1.1	7.7	1.9	30.8	**0**	**0**
NR	15	404	27054.5	3399.4	11326.8	4.7	**1.75**
PL	11.7	21.1	1721.4	571.9	2192.7	**0.2**	1.2
PO	0.1	2.2	6.9	2.2	37.1	**0**	**0**
RE	3.1	75.4	3163.3	265.6	627.2	**0**	1
SC	**0**	5.5	8.3	2	27.8	3.7	**0**
SE	**0**	1.8	7.6	4.2	49.1	**0**	**0**
ST	6.8	46.7	1344	303.2	1090.9	0.1	**0**
TE	**0**	**0**	0.3	**0**	24	**0**	**0**
UN	8.5	173.9	7365.2	1383.2	1877.3	7	**2**
VL	6.2	149.1	2455.6	439	1118.2	0.3	**0**
Mean	4.059	69.637	3934.93	607.574	1305.16	1.03704	**0.513**

Table 5. Relative Errors of Algorithms. Best results are typed with bold font (Domain is restricted

Dataset	RSBFP	RSBF	RFP	RPFP	KNN	RULE	MARS	DART
AI	0.593	0.641	0.499	**0.473**	0.612	0.744	0.720	0.546
BA	0.720	0.575	0.787	0.653	**0.443**	0.666	0.493	0.508
FA	0.720	0.379	0.887	0.667	0.785	0.820	**0.305**	0.638
HO	0.883	**0.715**	1.000	0.868	0.907	0.890	0.769	0.986
NO	0.969	0.975	1.040	**0.962**	1.232	1.250	1.012	1.112
NR	1.046	1.033	0.984	0.947	1.034	1.217	0.928	**0.873**
PL	0.833	**0.403**	0.895	0.415	0.475	0.477	0.404	0.432
ST	1.731	1.028	0.992	0.729	**0.599**	0.906	0.781	0.754
TE	4.577	4.697	4.014	**1.659**	1.895	4.195	7.203	2.690
Mean	1.341	1.161	1.233	**0.819**	0.887	1.241	1.402	0.949

Table 6. Train time of algorithms in milliseconds. Best results are typed with bold font (Domain is restricted)

Dataset	RSBFP	RSBF	RFP	RPFP	KNN	RULE	MARS	DART
AI	2.2	65	1	1.2	**0**	90.8	159.2	62
BA	24.4	590.7	14.9	21	**0**	181.8	915.1	3171.1
FA	19	607.3	11.4	16	**0**	161.1	985	1773.9
HO	11.6	444.9	6.5	10	**0**	57.5	616.3	893.9
NO	**0**	29.1	0.1	**0**	**0**	30.6	69.3	18.9
NR	173.7	1634.1	98	128.2	**7.4**	3493	5709.9	87815
PL	18.4	109.8	12.7	15.7	**0.2**	175.3	824.8	10024.4
ST	41.3	593.4	26.6	33.4	**1.4**	365.1	2281.4	17346.4
TE	**0**	50.7	**0**	**0**	**0**	30.9	31.1	3.1
Mean	32.289	458.333	19.022	25.056	**1**	509.567	1288.011	13456.522

Table 7. Test time of algorithms in milliseconds. Best results are typed with bold font (Domain is restricted)

Dataset	RSBFP	RSBF	RFP	RPFP	KNN	RULE	MARS	DART
AI	1	0.3	3.2	8.9	3.4	141.7	**0**	**0**
BA	2.3	2.1	29.4	298.3	54.6	244.8	**0**	**0**
FA	1.3	1.3	23.7	187.6	33.1	96.4	**0**	**0**
HO	0.4	**0**	10.9	104	13.3	43	**0**	**0**
NO	0.2	1.3	1.1	7.7	1.9	30.8	**0**	**0**
NR	15	11.8	404	27054.5	3399.4	11326.8	4.7	**1.75**
PL	11.7	8.6	21.1	1721.4	571.9	2192.7	**0.2**	1.2
ST	6.8	5	46.7	1344	303.2	1090.9	0.1	**0**
TE	**0**	0.1	**0**	0.3	**0**	24	**0**	**0**
Mean	4.3	3.389	60.011	3414.133	486.711	1687.9	0.556	**0.328**

References

[1] **Breiman, L, Friedman, J H, Olshen, R A and Stone, C J** '*Classification and Regression Trees'* Wadsworth, Belmont, California (1984)

[2] **Friedman, J H** 'Local Learning Based on Recursive Covering' Department of Statistics, Stanford University

[3] **Weiss, S and Indurkhya, N** ' Rule-based Machine Learning Methods for Functional Prediction' *Journal of Artificial Intelligence Research* Vol 3 (1995) pp 383-403

[4] **Aha, D, Kibler, D and Albert, M** 'Instance-based Learning Algorithms' *Machine Learning* Vol 6 (1991) pp 37 – 66

[5] **Quinlan, J R** 'Learning with Continuous Classes' *Proceedings AI'92* Adams and Sterling (Eds) Singapore (1992) pp 343-348

[6] **Bratko, I and Karalic A** 'First Order Regression' *Machine Learning* Vol 26 (1997) pp 147-176

[7] **Karalic, A** 'Employing Linear Regression in Regression Tree Leaves' *Proceedings of ECAI'92* Vienna, Austria, Bernd Newmann (Ed.) (1992) pp 440-441

[8] **Friedman, J H** 'Multivariate Adaptive Regression Splines' *The Annals of Statistics* Vol 19 No 1 (1991) pp 1-141

[9] **Breiman, L** 'Stacked Regressions' *Machine Learning* Vol 24 (1996) pp 49-64

[10] **Kibler, D, Aha D W and Albert, M K** 'Instance-based Prediction of Real-valued Attributes' *Comput. Intell.* Vol 5 (1989) pp 51-57

[11] **Guvenir, H A, Uysal, I** 'Regression on Feature Projections' *Proceedings of 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'99)*, Springer-Verlag, LNAI 1704, Jan. M. Zytkow and Jan Rauch (Eds.), Prague, Czech Republic, (September 15-18, 1999), 568-573.

[12] **Weiss, S and Indurkhya, N** 'Optimized Rule Induction' *IEEE Expert* Vol 8 No 6 (1993) pp 61-69

[13**] Friedman, J H**, Local Learning Based on Recursive Covering, 1996.

[14] **Uysal I ,** 'Instance-Based Regression By Partitioning Feature Projections', *M.S. Thesis*, Bilkent University, Ankara (2000)