Rule Extraction using Genetic Programming for Accurate Sales Forecasting

Rikard König
School of Business and IT
University of Borås
Borås, Sweden
rikard.konig@hb.se

Ulf Johansson
School of Business and IT
University of Borås
Borås, Sweden
ulf.johansson@hb.se

The purpose of this paper is to propose and evaluate a method for reducing the inherent tendency of genetic programming to overfit small and noisy data sets. In addition, the use of different optimization criteria for symbolic regression is demonstrated. The key idea is to reduce the risk of overfitting noise in the training data by introducing an intermediate predictive model in the process. More specifically, instead of directly evolving a genetic regression model based on labeled training data, the first step is to generate a highly accurate ensemble model. Since ensembles are very robust, the resulting predictions will contain less noise than the original data set. In the second step, an interpretable model is evolved, using the ensemble predictions, instead of the true labels, as the target variable. Experiments on 175 sales forecasting data sets, from one of Sweden's largest wholesale companies, show that the proposed technique obtained significantly better predictive performance, compared to both straightforward use of genetic programming and the standard MSP technique. Naturally, the level of improvement depends critically on the performance of the intermediate ensemble.

Keywords — *genetic programming, rule extraction, overfitting, regression, sales forecasting*

I. INTRODUCTION

Sales forecasting in the retailing domain is notoriously hard, especially for products that are frequently used in campaigns or promotions. First, the effect of a campaign is dependent of many factors not controlled by the company; e.g., weather and campaigning competitors. Second, a campaign is often more or less unique, in fact thousands of combinations of mechanisms (e.g., price, media channel offer types etc.) exist. A third problem is that there often are several simultaneous campaigns that interact. A fourth problem is the fact that campaigns often are run during holidays, making it particularly hard to discern the contribution of the campaign from the normal increased sales during a holiday. In practice, since the sales vary significantly from day to day, an aggregation to weekly sales are often necessary to find any relevant patterns. This, however, of course limits the number of data points that can be used for learning a predictive model. Even if the company is fortunate enough to have gathered data over a period as long as three years, the result is a tiny data set consisting of approximately 150 examples. The lack of examples to learn from, especially when recognizing the relatively high dimensionality of the data sets, makes the risk of overfitting apparent.

Sales forecasting is a special case of predictive modeling, where the goal is to make predictions about some future phenomenon (here sales) based on previous observations. More

technically, the purpose of predictive modeling is to use labeled data and supervised learning to create a model representing the relationship between the independent (input) variables and the dependent (output or target) variable.

Genetic Programming (GP) has traditionally not been widely used for predictive modeling, at least not in comparison to typical predictive techniques like decision trees or artificial neural networks. There are, however, a small number of successful applications see e.g., [1]-[3]. Still, we argue that GP is in fact remarkably well suited for predictive tasks. In [4], it is noted that GP has three advantages when used for predictive modeling; i) the fitness function can be designed identically to the criterion used for the evaluation, ii) the comprehensibility can be increased by tailoring the representation based on the preferences of a decision maker and iii) the global optimization should increase the predictive performance. In addition, Koza in [5] also showed how a parsimony pressure can easily be added to the fitness function, which in essence is a basic way of handling the accuracy vs. comprehensibility trade-off. Based on the description above, we argue that the most important problem that must be solved in order to recognize GP as a strong and robust predictive technique is the inherent tendency to overfit the training data.

With this in mind, the main purpose of this paper is to suggest and evaluate a novel method for GP regression, in situations where interpretable models are mandatory, explicitly reducing the risk of overfitting small and noisy data sets. Another purpose is to demonstrate how GP-based regression models may benefit from the possibility to pick and change different optimization criteria.

II. BACKGROUND

A. Scoring in Predictive Modeling

Predictive modeling, as depicted in Figure 1, can be performed using many different techniques, but most of them optimize some score function over the training data set. Most typically, the prediction error is minimized. Once generated (or trained) the resulting model can be used to predict the value of the dependent variable for novel instances.

Considering that so many score functions exists, and that no score functions is optimal for all predictive problems, it is surprising that most predictive techniques used today are still restricted to a single predefined score function. GP which is the technique used in this paper is one of the few exceptions since arbitrary score function can be implemented in the fitness function. It can also be argued that the predefined score function

This work was supported by the Swedish Retail and Wholesale Development Council through the project Innovative Business Intelligence Tools (2013:5).

used internally by the traditional predictive technique, most often differs from the optimization function presented by the actual problem. An obvious example is when some kind of greedy approach is used to induce decision trees. Here, local choices are made based on some information theoretic criterion, like purity or information gain, while the overall (global) criterion is classification accuracy. Naturally, it would, in many situations, be preferable to use a technique which allows the data miner to choose a score function directly related to the overall goal of the current project. In most modeling techniques, though, the score function is fixed and an integral part of the learning algorithm. For example, regression modeling techniques like multiple linear regression, Auto Regressive Moving Average and Exponential smoothing all minimize the mean square error (MSE), while regression trees such as M5 [6], may use the standard deviation as optimization criterion during the construction and the root mean square error (RMSE) during pruning.

Fig. 1. Overview of predictive modeling

An important property often mentioned as the main reason for using these criteria is that they penalize larger errors more severely. RMSE or MSE are, however, far from obvious choices. In fact, Armstrong [7], even argues that they are some of the worst metrics since they are poorly protected against outliers while being scale-dependent, i.e., errors from two data sets cannot be compared if the dependent variables of the data sets do not have the same scale.

Accuracy and comprehensibility are often mentioned as two important criteria for predictive models. A drawback for most comprehensible models is that they in general are less accurate than opaque models, i.e., models that are not interpretable due to their representation language or complexity. It is, for example, well known that ensemble models generally are more accurate and robust than single model of the same type, see i.e., [8]. Unfortunately, all ensembles are inherently opaque, i.e., they cannot be analyzed or used to explain a specific prediction, even if the included base models are comprehensible in the first place. The choice of using a more accurate opaque model or a less accurate comprehensible model is therefore a common dilemma within predictive modeling, often referred to as the accuracy vs. comprehensibility trade-off, see e.g., [9],[10]. In many decision support scenarios, comprehensible models are mandatory, since they can be forced to, at the very least, explain the basis for a particular decision.

B. Forecasting using genetic programming

As mentioned in the introduction, GP has several advantages compared to traditional predictive modeling techniques. However, straightforward GP approaches often strug-

gle with symbolic regression problems. Keijzer, in [11], argues that the main problem is that the selection pressure enforces the GP process to spend most of its effort on getting the range of the constants right. Once found, the population diversity is often so low that the rest of the expression is never found. However, Keijzer also shows that this problem, called *the range problem*, can be eliminated for polynomial expressions by scaling the output of each program with a simple linear regression. Given that p_i is the prediction of the program for the instance i the prediction can be scaled using equation (1), where the slope k and the intercept m for the linear regression are calculated in advance, using the least square method.

$$p_{i_{scaled}} = k * p_i + m \quad (1)$$

In regression trees, the same problem of finding the correct range appears in each and every leaf node. Unfortunately, it does not help to scale the final output of the model, simply because the range of the intercept would need to be at a scale appropriate for the associated variable.

For the regression tree representation, which is the focus of this study, one straightforward solution would be to use the average value of all training instances reaching a specific leaf as the leaf constant. The resulting average value would most often, however, only be optimal if RMSE was selected as the criterion to minimize. For other metrics such as mean absolute error (MAE) or mean absolute percentage error (MAPE) a different leaf value would most likely be the optimal choice.

Consider for example a leaf node which observes three training instance with the target values 1, 2 and 4; the average value 2.33 would result in the minimum error according to RMSE. However, as can be seen in table I, it would, according to MAE, be better to predict the value 2, while a prediction of 1 would be optimal for MAPE.

TABLE I. SIMPLE OPTIMIZATION EXAMPLE

Observed Values	Predicted Value	MAE	RMSE	MAPE
[1, 2, 4]	2.33	1.11	1.25	82.1%
[1, 2, 4]	2	1.00	1.29	64.3%
[1, 2, 4]	1	1.33	1.83	53.6%

As demonstrated by this simple example, using mean values as leaf constants is only optimal when using RMSE. Or, put the other way around, the method to find good ephemeral constants are dependent on the optimization criterion, and specifically, using mean values can actually be suboptimal if any other metric than RMSE is used.

Another more advanced approach was used in [12], where regression trees were evolved using a fitness function based on MAE. The initial population was created using an algorithm similar to CART. To achieve diversity, each individual was created using randomly selected subsets of the training data. Since the ephemeral constants were created using the decision tree algorithm, they would always be in an appropriate range. In that study, mutation was also performed by calculating a new split with the same method. Finally it should be noted that this and similar methods automatically make the constants dependent on the criteria optimized by the decision tree algorithm, e.g., least square or least deviation. So, the constants were initialized based on RMSE, the tree was optimized on MAE, and the final model was evaluated on RMSE. Clearly,

the implicit choice to mix these error metrics, especially evaluating on a metric not targeted during the model construction, may very well be the reason for the somewhat discouraging results.

However, a previous study [13], demonstrated that a simpler and more straightforward approach could suffice. In that study, all leaf constants C_i were initialized to a random value between zero and one. These internal constants were scaled by the maximum and minimum value of the dependent variable y for the training instance reaching the leaf l according to equation (2).

$$C_{rnd} = \min(y_l) * C_i * + (1 - C_i) * \max(y_l) \quad (2)$$

This approach drastically reduced the range problem while, at the same time, allowing optimization of an arbitrary fitness function. Another advantage is that the general meaning of the internal constants is conserved through crossover, i.e., a high value will result in a prediction near $\max(y_l)$ in all leaves, even if $\max(y_l)$ will be different in different leaves.

C. Forecasting using ensembles

The most intuitive explanation for why ensembles work is that the aggregation of several models, using averaging or majority voting, will eliminate uncorrelated base classifier errors. Thus, a key property of effective ensemble techniques is that the selected base models are sufficiently diverse, ideally distributing errors evenly over the base models. Bagging [14], which is one of the most common ensemble creation methods, obtains diversity by using resampling to create different training sets for each base classifier.

As mentioned above, the main drawback of using ensemble learning is that the interpretability is lost. Several researchers have suggested rule extraction algorithms, transforming opaque models like neural networks, support vector machines or ensembles into comprehensible models, keeping an acceptable accuracy, see e.g., [15] and [16]. In *pedagogical rule extraction* a new comprehensible model is created based on the predictions of the ensemble. More precisely, a new training dataset is formed by replacing the target value with the prediction of the ensemble. Next, standard supervised learning is used to generate a model representing the relationship between the input and the opaque model predictions.

D. Handling Noisy Data

Teng, in [17], lists three main approaches to handling noise within in the field of machine learning:

- avoiding overfitting by some kind of pruning or stopping criteria for training, to prevent construction of overly complex models.
- removing noisy instances from the training data based on some evaluation mechanism.
- correcting noisy instances using some correction mechanism.

Handling noise by avoiding overfitting is of course the standard approach where most techniques have some kind of functionality; M5, as an example, prunes the overly complex trees based on RMSE [6]. GP does instead use some kind of parsimony pressure to favor smaller and more general rules during evolution. A problem with these and similar techniques is that it is hard to set the right amount of pruning / pressure, often resulting in a too general or too complex model. With this in mind, a second validation set may be used to find the

right amount of generalization. However, this is not feasible for small datasets since it requires that a large part of the precious training instances will not be available for training.

Ensembles were used in [18] by Brodley and Friedl to identify and remove noisy instances. Results showed that ensembles were better at identifying noisy instances compared to single algorithms. Furthermore, ensembles using consensus voting were conservative at throwing away good data at the expense of retaining noisy data and majority voting were better at detecting noisy data at the expense of throwing away good data. Other approaches to removal instances containing noisy data include [19] and [20]. However if small datasets are considered, removing data is not a practical solutions. Hence, considering correcting noisy instances is the most feasible approach.

In [17], Teng presents an interesting approach that aims to correct both noisy attribute values and mislabeled class values. First ten C4.5 trees are created and evaluated for the training data using 10-fold cross validation. Next, the proposed polishing technique is applied to the missclassified instances. Attributes is first polished by switching the data sets, thus using the noisy attribute as the target and the target variable and all other attributes as input. Ten trees are again built for each attribute and are then used to replace the noisy values of the attribute. If no attribute change results in a correct prediction of the attribute, the class attribute is replaced with the predicted values. This procedure is repeated until the whole dataset can be predicted correctly. Experiments showed that decision trees built on the polished data became both more accurate and comprehensible. The main drawbacks of this technique, noted by the authors, are that is time consuming and can only be applied to nominal attributes. In addition, it assumes that the attributes are somehow related.

III. METHOD

The general idea of the proposed method, described in Figure 2, is to remove noise from the training data by replacing the correct target values with the predictions of an ensemble, i.e., in fact by utilizing pedagogical rule extraction as an interior tool. Here, however, the aim is not to explain the predictions of the ensemble, but to reduce the risk of overfitting. Since ensembles generalize well, their predictions in general are smoother, i.e., contain less noise than the original data. Naturally, if the data contains less noise there is less danger of overfitting. This means that the predictive model can be trained as long as necessary to model the training data accurately.

The proposed method is similar to both the filtering approach, suggested by Brodley and Friedl, and the correction technique presented by Teng, since an ensembles is used as the tool for identifying and correcting noisy data. It differs from the filtering approach by not eliminating data and to the polishing technique by only considering the target variable for corrections. Another important difference is, of course, that the problem here concerns forecasting, which cannot be handled by the mentioned techniques. Since prediction of a continuous variable is a much harder problem than classification the result is that all trainings instances is more or less corrected by the ensemble.

Fig. 2. Overview of the proposed method

In the experiments, an ensemble of 20 M5P model trees is first created using bagging. Pedagogical rule extraction is then applied by simply correcting/exchanging the actual value of the training instances with the prediction of the ensemble. It is important to note that the ensemble is only used during training of the interpretable model and does not need to be used when predicting new instances. Hence, the evolved trees remain comprehensible in spite of the fact that they are trained using the ensemble.

A. Evolving regression trees

When the actual value has been replaced by the ensemble prediction, a regression tree is evolved using a specific optimization criterion using G-REX, an open source GP framework for data mining [21]. The error measures targeted in this study, MAE, RMSE, MAPE and R , have been implemented into four different fitness functions. The general form of these fitness functions is given in Equation 3. Here, M is the optimized measure, p is the program under evaluation, O_p is the complexity of p (nodes + leaves) and P is the coefficient for the parsimony pressure.

$$f_{M_p} = M_p + P * O_p \quad (3)$$

MAPE, and R are scale-independent, while MAE and RMSE can be made scale-independent by simply dividing by the mean of the actual value. Scale independence of course ensures that the same parsimony pressure will have a similar effect for different data sets, thus minimizing the amount of parameter tuning. Still, the effect may vary slightly since each metric represents a different error function, i.e., the change in deviation does rarely affect the error identically for all measures. Finally R values were replaced with $1-R$ to facilitate minimization of all fitness functions. Since, the idea is to allow optimization of different metrics the predictions in a leaf is calculated according to equation 2.

IV. EXPERIMENTS

In this study, two main experiments will be performed. First the generalization ability will be evaluated by optimization of a program for each metric. These programs will then be compared against M5P model trees, regression trees and an ensemble of bagged M5P model trees. However, since the assumption is that the problem domain requires comprehensible models, M5P regression trees are used as the main performance benchmark. The second experiment will use identical settings, except that the programs will now be extracted from an ensemble, i.e., the purpose is to investigate if this procedure

will reduce the overfitting, thus leading to higher accuracy. It must be noted that the regression trees evolved using G-REX have the same representation as the regression trees produced by M5P, i.e., with constants in the leaves.

All models, regardless of which technique that was used to create it, are evaluated against four common error metrics; MAE, RMSE, MAPE, and R . Since MAPE may run into problems when the target values are close to zero, they are trimmed by setting the maximum error for a single instance to 10.

Experiments are performed on 175 products from one of Sweden's largest whole-sale companies. The predictions task is not a typical sale forecasting problem since it concerns the amount ordered from the central stock by the stores. Store owners are franchisers but can decide themselves how much they want to order. Since the sales during campaigns are especially hard to predict, the chosen products are a stratified subset of all products having at least four campaigns per year. About three years of weekly sales and campaign data are available for each product. The forecast horizon is two weeks and four weeks of lagged input were created for each input variable to convert the problem into a typical regression (time series forecasting) problem. Input variables contained data about price, discount, offer mechanic, campaign media type and campaign coverage i.e., the number of stores that participated in the campaign. For the actual evaluation, 75% of the data was used for training and the last 25% was used as the test set.

In this study, G-REX creates regression trees, using the BNF defined in table II, for each of the four fitness functions, as described above.

TABLE II. BNF FOR THE EVOLVED REGRESSION TREES

<i>Functions</i>	{ if, <=, >, = }
<i>Terminals</i>	{ catV, conV, C _{mdl} , C _{conV} , C _{catV} }
<i>program</i>	::= expression
<i>expression</i>	::= if-statement / C _{mdl}
<i>if-statement</i>	::= if condition then expression else expression
<i>condition</i>	::= conV <= C _{conV} / conV > C _{conV} catV = C _{catV}
<i>C_{mdl}</i>	::= Constant initialized according to equation 2.
<i>C_{conV}</i>	::= Constant initialized to a random value of conV
<i>C_{catV}</i>	::= Constant initialized to a random value of catV
<i>conV</i>	::= Independent continuous variable
<i>catV</i>	::= Independent categorical variable

To achieve more robust results, a batch of three separate runs was performed for each data set. The winning tree with the best fitness from the three batches was selected as the final tree, and consequently used for evaluation on the test set. Since the optimization criteria encoded in the fitness functions were made scale independent and equal in range, the same set of parameters could be used for all fitness functions, see table III. These parameters, which are typical default settings for GP, were deemed appropriate based on initial experiments on five dataset not used in the main experiments.

TABLE III. GP SETTINGS

GP Parameter	Value
Population Size	500
Generations	50
Creation Type	Ramped Half and Half
Crossover Probability	0.8
Mutation Probability	0.001
Parsimony pressure (P)	0.01
Batches	3

V. RESULTS

Table IV below shows the results for the three M5P based techniques. The results are the average results for the 175 data sets, the best result given in bold. First, it is interesting to note that the results are quite different for the different metrics. Obviously, the different techniques have properties that apply better to some evaluation criterion, even if they optimize the same measure. Compared to a single M5P tree, the ensemble M5P-Bag is clearly better in terms of MAE, RMSE and R but actually worse in terms of MAPE. It is also interesting to note that regression trees created using M5P-Reg is performing comparable to M5PBag according to all metrics except R , in spite of the much simpler (and more comprehensible) representation language. Furthermore, M5P-Reg, like the ensemble, outperforms M5P in all metrics but MAPE.

TABLE IV. COMPARING TECHNIQUES

Technique	MAE	RMSE	MAPE	1-R
M5P-Bag	22.6	29.7	40.2	36.8
M5P	30.1	49.6	30.0	64.0
M5P-Reg	24.2	33.3	41.2	51.4

The fact that M5P-Reg performs so much better than M5P on MAPE can be explained by overfitting the quite noisy data sets. However, M5P seems to make low forecasts, possibly focusing on non-campaign weeks, since it is outstanding in regards of MAPE, which is biased to this type of forecast. M5P-bag does, however, perform well since it combines 20 M5P trees based on different bootstraps of the data set. In an ensemble created using bagging, overfitting is rarely a big problem, since each M5P tree is trained on a subset of the data set. At least in theory, each tree will overfit different aspects of the data set, so their errors should ideally cancel out. Finally, the high amount of noise in the data sets can be seen from the fact that M5P-Reg performs almost as well as the ensemble, except in terms of R , in spite of the much simpler regression tree representation. The low results regarding R hints of baseline prediction, close to the average value, which does not follow the changes in the target variable as well.

The results regarding the creation of regression trees using a standard GP approach are presented in the next two tables below. Table V presents the training performance in order to evaluate if the optimization was successful. Again the best result is given in bold.

TABLE V. GP TRAINING ERROR

	MAE	RMSE	MAPE	R
f_{MAE}	16.1	23.2	18.3	25.1
f_{RMSE}	16.4	21.8	19.7	23.2
f_{MAPE}	17.6	27.1	16.9	28.4
f_R	20.1	26	22.6	17.7

Obviously, the optimization using different metrics succeeds in the way that the best training score for each metric is actually achieved when the same metric is optimized in the fitness function. However, Table VI shows that this does not always carry over to novel data, probably due to the noisy nature of the data sets. This is clearly demonstrated by the test results which show that the trees optimized for MAPE actually outperforms all other optimization criteria regardless of evaluation metric. Interestingly enough, M5P-Reg which is included

as a benchmark, is also outperformed by the trees optimized for MAPE, but performs reasonable well overall.

TABLE VI. GP TEST ERROR

Technique	MAE	RMSE	MAPE	1-R	SIZE
M5P-Reg	24.2	33.3	41.2	51.4	8.5
f_{MAE}	25.3	34.3	42.2	48.9	13.2
f_{RMSE}	25.1	34.5	42.6	46	12.8
f_{MAPE}	23.7	32.8	37.3	45.1	11.9
f_R	28.5	38.1	43.8	45.3	16.6

Except for the result regarding MAPE, there is no obvious pattern indicating how the techniques perform, even when considering the results on the training data. The most likely explanation is overfitting, which is to be expected when applying GP to small and noisy data sets. The size of the generated trees, reported in Table VI as the number of nodes and leaves, are comparable and all must be considered to be comprehensible.

Table VII below presents the results when the programs were extracted from an M5P-Bag ensemble. Underlined results indicate that the rule extraction improved the performance, and the best results are again given in bold.

TABLE VII. GP EXTRACTION TEST ERROR

Technique	MAE	RMSE	MAPE	1-R	SIZE
M5P-Reg	24.2	33.3	41.2	51.4	8.5
Ef_{MAE}	<u>24.0</u>	32.0	41.8	46.0	14.4
Ef_{RMSE}	<u>24.2</u>	<u>31.8</u>	43.0	44.6	14.0
Ef_{MAPE}	24.4	32.4	42.0	47.6	13.8
Ef_R	<u>24.2</u>	31.9	<u>40.6</u>	<u>43.8</u>	14.5

Here, the results are more what a data miner would hope for i.e., that the best result for each metric was actually achieved by using the corresponding fitness function. The only exception is MAPE. It is, however, not surprising that the M5P-Bag extraction did not improve the performance for Ef_{MAPE} , since the ensemble model was not that much more accurate to start with. In fact, the MAPE result for the M5P-bag ensemble was only marginally better than M5P-Reg, and even the trees produced by f_{MAPE} . Of course there are no really big differences between the different fitness functions except in terms of correlation but this is natural since the same M5P-bag ensemble optimized SE i.e. (variance and RMSE) was used in combination with all fitness functions. The size of the extracted programs were slightly larger than the original GP technique but still small and comprehensible.

In the end only one optimization criterion can be used for a particular problem and the result correlate with the intuitive choice of optimizing the same metric that is used for the evaluation. Hence, this approach is evaluated using a pairwise comparison while considering GP (f_x) and extracted GP (Ef_x) as predictive techniques. Each column in table VIII shows the difference in the evaluated metric (in favor for the first technique) and the p-values for a Wilcoxon signed rank test over all 175 datasets.

TABLE VIII. PAIR WISE COMPARISON OF TECHNIQUES

X	Ef_X vs f_X		f_X vs M5P-Reg		Ef_X vs M5P-Reg	
	diff	p	diff	p	diff	p
MAE	1.3	$2.2e^{-2}$	-1.1	$4.0e^{-2}$	0.2	$8.8e^{-1}$
RMSE	2.7	$1.5e^{-5}$	-1.2	$4.3e^{-2}$	1.5	$1.3e^{-2}$
MAPE	-4.7	$3.2e^{-4}$	3.9	$1.0e^{-2}$	-0.6	$4.7e^{-2}$
R	1.5	$6.7e^{-1}$	6.1	$7.4e^{-3}$	7.6	$8.7e^{-4}$

The comparison between f_X and Ef_X shows that the rule extraction approach is significantly better in terms of MAE and RMSE but significantly worse for MAPE. Surprisingly, the result for Ef_R is not significant even if the improvement after the extraction is larger than for Ef_{MAE} for which the results is significant.

However, a closer examination of the results shows that Ef_R only outperforms f_R on 54% of the data sets and that f_R have rather bad results for a few data sets which make the average difference look bigger.

Compared to M5P-Reg, the normal GP approach is significantly better in terms of R and MAPE, but significantly worse with regard to MAE and RMSE; which is the optimization criterion used by M5P. Ef_X is significantly better than M5P-Reg in terms of RMSE and R , and is only outperformed in regards of MAPE which again is a natural result since M5P-Reg had a lower MAPE than the ensemble which the rules were extracted from. Even if Ef_X is not significantly better than M5P-Reg in regards of MAE the approach clearly works since M5P-Reg was significantly better than the traditional approach f_X .

The main result of the experiments is that the suggested approach of using an intermediate model to reduce the noise in the training data works remarkably well – but only when the ensemble itself is a highly accurate model. Hence, it is interesting to analyze in detail how the rule extraction from an ensemble modifies the prediction task. The charts below are selected to demonstrate typical effects of the ensemble rule extraction approach. Two articles with different sales pattern are analyzed; product A has a typical seasonal pattern and product B has frequent campaigns and no typical seasonal patterns. The actual sales are shown using a red solid line while the prediction of f_{MAE} is shown with a blue dotted line. Campaigns are shown with a solid black line along the x-axis. Key values for the actual sales are marked using an a , predictions with a p , prediction based on rule extraction with and x and campaigns with a c .

Figure 3 shows the predictions produced by f_{MAE} on both the training- and test set. The data has three typical spikes which occur at the same holiday each year i.e., $a1$, $a2$ and $a3$. f_{MAE} has clearly learned the training set and its two holiday events, even if the prediction $p1$ for the second occurrence $a2$ is somewhat low. Smaller variations between the holidays have also been identified rather accurately. On the test set, however, the actual holiday $a3$, which occurred in the first instance in the test set is severely underestimated at $p2$. The predictions $p3$ and $p4$ for two of the following weeks are instead greatly overestimated. In this case, the over- and underestimates can be explained by a mix-up of event and campaign effects. The first holiday $a1$ in the chart has been credited to the campaign $c1$ two weeks earlier and $p3$ and $p4$ are the effects of the same type of campaign $c2$, $c3$, which again appear two weeks before each prediction.

Fig. 3. Normal forecast for Product A

Figure 4 instead shows the effect of applying the rule extraction on the same dataset. Hence, the filled red line represents the predictions of the ensemble for the training data while, in the test set, the filled red line still represents the actual value. Key ensemble predictions are marked using e while the extracted predictions of Ef_{MAE} are marked using x . $a3$ in the test set is of course still the actual value.

Fig. 4. Extracted forecast for Product A

From this chart, it is clear that using ensemble predictions has dampened the variation between the holidays making it easier to find a good baseline for this period. A more important result, however, is that $e1$ and $e2$ are of much more similar height compared to the actual values $a1$ and $a2$, corresponding to the two holidays in the original data presented in the figure 3. Hence, the ensemble predictions for the holidays are more easily recognized as a recurring event rather than as a successful campaign. The result is that Ef_{MAE} makes the same prediction for $x1, x2$ and $x3$ and does not make the same erroneous estimations based on $c2$ and $c3$ which f_{MAE} did in figure 3. Even if $x3$ is a bit lower than $a3$ it is a much better estimation than the disastrous $p2$ in figure 3.

Next, product B , which is much more frequently campaigned and does not show any seasonal or holiday patterns, is analyzed in the Figure 5 below. It must be noted, that there are almost as many campaigns that have a large effect on the sales as campaigns that seem to have no effect at all. Again there are two larger spikes ($a8$ and $a9$) at the end of the training set, but these are predicted rather accurately. Here the problem is instead that f_{MAE} does not find the correct estimates for $a1-a7$ resulting in too low estimates for $a10-a14$ in the test set.

Fig. 5. Normal forecast for Product A

Next, Figure 6 shows the results of the rule extraction, using Ef_{MAE} , for the same product. Here the result of the extraction is apparent from the ensemble predictions $e1-e9$ that are of more similar height and hence easier to learn. More importantly, Ef_{MAE} makes very accurate prediction for $e1-e9$, which also results in more accurate predictions for $a10$, $a12$ and $a13$. The predictions are not as good for $a10$ and $a14$, but they are still much more accurate than the prediction made by f_{MAE} in figure 5.

Fig. 6. Extracted forecast for Product B

It should be noted that these data sets were selected since there was a clear benefit of using the rule extraction approach. Of course the results were not as clear for all data sets, even if Ef_{MAE} and Ef_{RMSE} were significantly better overall than their normal GP counterpart.

VI. CONCLUSIONS

One purpose of this study was to demonstrate that GP is suitable for predictive modeling. One important reason is the ability to optimize the actual score function instead of using a predefined score function that may not be optimal for the problem at hand. Experiments showed that GP was capable of optimizing MAE, RMSE, MAPE and R, on training data, but that the result rarely carried over to the test set due to the noisy characteristic of the small data sets typical for the sales forecasting domain.

However, when the proposed rule extraction was applied, the correlation between performance on the training set and the test set was much higher. More specifically, a model optimized for a specific metric on the training data was also superior on the test set, compared to models optimized using other metrics.

In a pairwise comparison the extraction approach was superior on three of the four metrics used for evaluation. When compared using MAPE, however, the extraction approach was significantly worse – a result explained by the modest ensemble performance. The extraction approach was also superior to regression trees produced using M5P. When the same metric was used for optimization as for score function the regression trees were clearly outperformed on all metrics but MAPE, again for the same reason as discussed above.

REFERENCES

- [1] A. Agapitos, M. Dyson, J. Kovalchuk, S. M. Lucas, On the genetic programming of time-series predictors for supply chain management, *Proceedings of the 10th annual conference on Genetic and evolutionary computation* 2008
- [2] A. L. Garcia-Almanza and E. P. K. Tsang. Forecasting stock prices using genetic programming and chance discovery. In *12th International Conference On Computing In Economics And Finance*, 2006.
- [3] H. Iba and N. Nikolaev. Genetic programming polynomial models of financial data series. In *Proceedings of the IEEE 2000 Congress on Evolutionary Computation*, 2000.
- [4] P. G. Espejo, S. Ventura, and F. Herrera, “A Survey on the Application of Genetic Programming to Classification,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 40, no. 2, pp. 121–144, 2010.
- [5] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection*. Cambridge, Massachusetts: MIT Press, 1992.
- [6] J.R. Quinlan, Learning with continuous classes. In 5th Australian joint conference on artificial intelligence. pp. 343–348, 1992.
- [7] J. S. Armstrong, *Principles of forecasting: a handbook for researchers and practitioners*. Boston, Mass. London: Kluwer Academic Publishers, 2001.
- [8] T. G. Dietterich, “Machine learning research: four directions,” *The AI Magazine*, vol. 18, 1997.
- [9] P. Domingos, “Knowledge discovery via multiple models,” *Intelligent Data Analysis*, pp. 1–18, 1998.
- [10] V. R. Blanco, O. J. Hernandez, and Q. M. Ramirez, “Analysing the trade-off between comprehensibility and accuracy in mimetic models,” *Lecture Notes in Computer Science*, vol. 3245, pp. 1–8, 2004.
- [11] M. Keijzer, “Improving symbolic regression with interval arithmetic and linear scaling,” *Genetic Programming*, pp. 275–299, 2003.
- [12] M. Kretowski and M. Czajkowski, “An evolutionary algorithm for global induction of regression trees,” in *Artificial Intelligence and Soft Computing: Part II*, 2010, pp. 157–164.
- [13] R. König, U. Johansson, and L. Niklasson, “Optimization and Evaluation Criteria in GP Regression,” in *International Conference on Data Mining (DMIN)*, 2012.
- [14] Breiman, L. 1996. Bagging predictors. *Machine Learning*, 24(2), pp.123–14.
- [15] M. W. Craven and J. Shavlik, “Understanding time-series networks: A case study in rule extraction,” *International Journal of Neural Systems*, 1997.
- [16] U. Johansson, C. Sönström, R. König, and L. Niklasson, “Neural networks and rule extraction for prediction and explanation in the marketing domain,” in *International Joint Conference on Neural Networks*, 2003, pp. 2866–2871.
- [17] Teng, C. M. Correcting noisy data, in *Proceedings of 16th International Conference on Machine Learning*, pp. 239–248, 1999.
- [18] Brodley, C. E., Friedl, M., Identifying and mislabeled training data, *Journal of Artificial Intelligence Research*. 11, 1999.
- [19] Brodley, C. E., Friedl, M., Identifying and eliminating mislabeled training instances, In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.
- [20] George, H. J. Robust decision trees: Removing outliers from databases, In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pp. 174–179, 1995.
- [21] R. König, U. Johansson, and L. Niklasson, “G-REX: A Versatile Framework for Evolutionary Data Mining,” in *IEEE International Conference on Data Mining Workshops, 2008. ICDMW’08*, 2008.