Direct Domain Knowledge Inclusion in the PA3 Rule Induction Algorithm

Pedro de Almeida

CISUC – Centro de Informática e Sistemas da Universidade de Coimbra,
Polo II da Universidade de Coimbra, 3030 Coimbra, Portugal
Physics Department, Universidade da Beira Interior, 6200 Covilhã, Portugal
nop00997@mail.telepac.pt

Abstract. Inclusion of domain knowledge in a process of knowledge discovery in databases is a complex but very important part of successful knowledge discovery solutions. In real-life data mining development, non-structured domain knowledge involvement in the data preparation phase and in the final interpretation/evaluation phase tends to dominate. This paper presents an experiment of direct domain knowledge integration in the algorithm that will search for interesting patterns in the data. In the context of stock market prediction work, a recent rule induction algorithm, PA3, was adapted to include domain theories directly in the internal rule development. Tests performed over several Portuguese stocks show a significant increase in prediction performance over the same process using the standard version of PA3. We believe that a similar methodology can be applied to other symbolic induction algorithms and in other working domains to improve the efficiency of prediction (or classification) in knowledge-intensive data mining tasks.

1 Introduction

In most cases, the availability and the efficient use of Domain Knowledge (DK) during the development process of a Knowledge Discovery in Databases (KDD) system is essential for successful knowledge discovery. In fact, DK is needed for almost any practical knowledge discovery task, independently of the domain or of the data mining techniques used, since, at least, some form of DK must be involved in the problem definition, in the data preparation and in the results evaluation and utilization phases. Sometimes, however, the involvement of DK in the process does not result in all the advantages it could bring. In fact, in some real-life situations where KDD could be useful, the available formally specified DK is restricted to description or definition of data and other forms of DK (for example theories about the way domain variables interact) exist only in informal, sometimes uncertain, non-structured forms. This kind of limitation of previously existing DK, together with a somewhat scarce theoretical work on the topic, usually results in no deliberate involvement of existing DK in the specific data mining phase of many real-life KDD processes.

DK involvement in the data mining step of a KDD process always implies a conditioning of the search of hypotheses conducted by the data mining algorithm. This conditioning can operate through an “initialization bias” (introducing starting conditions for the search), or through a “search bias” (distorting the search space, or the evaluation of hypotheses) [14], [15].

DK can be included in the data mining phase through direct integration (implicit or explicit) in the data mining algorithm, or through an associated knowledge base. In the first case, specific changes to the core data mining algorithm must be performed, in order to directly represent the involved domain knowledge through a biasing of the search. In the latter case, a very tight coupling between the domain theory description in the knowledge base and the bias representation language accepted by the learner is need, eventually involving an intermediate knowledge "translator" [3]. Anyway, both of these forms of DK integration tend to need software specifically adapted for each application case, since different kinds of domain knowledge usually involve different representations, and most data mining algorithms (and commercial data mining programs) don't allow the integration any form of DK not contained in the data.

Direct integration of DK in data mining software generally intends to direct and focus the pattern search that takes place at that KDD step. This can raise another potential limitation of this technique: If badly directed, the focused search can miss some of the potentially interesting patterns that an unbiased search could find in the data [4]. However, in spite of the limitations and potential problems, we believe that, in some cases, careful DK integration in the data mining step of a KDD process can produce significant improvements in the overall efficiency of the process.

This paper presents an experiment that integrates two domain theories directly in a rule induction data mining algorithm. The domain is short-term stock market prediction, and the two theories bias the algorithm, during rule search, against a specific class of rules, and towards another. The theories are tested over five data sets that correspond to multivariate information based on daily quotes of five of the most significant stocks in the Portuguese BVLP stock exchange. The base rule induction algorithm used, PA3 [1], is a recent general-purpose sequential cover algorithm that combines general-to-specific and specific-to-general search to develop each rule.

2 Domain knowledge

Adopting a restrictive DK definition, we will be interested only in domain theories that explain or predict future behavior of stocks on the basis of known data. This kind of domain theory is extremely uncertain in stock market prediction. There are, basically, three different positions: Those who believe that the markets are highly efficient and, as a result, essentially unpredictable, those who advocate "fundamental analysis" of the business results of the quoted companies, and those who believe that "technical analysis" (the analysis of historical stock quotes data, isolated of other known facts) is enough to predict the future behavior of those stocks [6].

The "efficient market" hypothesis, at least in its weakest form, has been traditionally accepted in some academic circles as basically correct, and if that were really the case, any effort to predict future behavior of listed stocks would be futile. However, besides the firm belief of those who really invest in stock markets (most of the investors and all the speculators), there is a growing body of published research indicating that at least some markets exhibit imperfections (which translate to a degree of predictivity) [7], [16], [11].

Classic "fundamental analysis" has solid background theory but even when successful in the long term, is not very useful to predict short-term movements of stock values [7]. A marginal aspect related to fundamental analysis that can be linked to very important fast movements of stock prices is the announcement of surprising fundamental company information (or surprising macroeconomic information, relevant for the whole market). However, this kind of fast readjustment of fundamental expectations will not be explicitly integrated in the analysis conducted in this paper, since it does not seem relevant for the paper's objectives and it requires very complex base data, and very demanding data preparation.

The theory behind present "technical analysis" is abundant. Unfortunately it is also fragmented and many times of dubious quality, most of it corresponding to unproved, sometimes untested, hypotheses. Moreover, the fact that technical analysis theory is still not seriously established can hide a fundamental problem: Even if technical analysis is realistically possible, perhaps it cannot be generalized for different markets, or for different stocks and different time frames of a market.

3 The problem and the data

The work we are involved in aims to predict the future behavior of five stocks listed in the Portuguese BVL stock exchange, utilizing historical data and DK.

This paper describes work done on direct domain theory integration in a rule induction algorithm used for the prediction of the next day behavior of each stock (binary prediction of rise or fall). This kind of next-day prediction is not enough to develop an operational trading strategy, but it is frequently found in the literature [2], [9], [11], and seems adequate to test the validity of the two domain theories involved.

For this very short-term prediction task, we simplified the base data by omitting fundamental information (and by not accounting for dividend payments), and used only historical stock quotes, transaction volumes and index values. It should be noticed that this base data has low information content for the prediction task, and could never result in very high accuracy rates, even with ideal data preparation and data mining steps. This situation is similar to having very noisy data both for learning and testing, and tends to present overfitting problems during the data mining process. With this problem in mind, we selected the domain theories to integrate in the rule-induction software aiming to reduce overfitting of the training data.

The five companies chosen for prediction are among those more actively traded in the BVL stock exchange: BCP, Brisa, Cimpor, EDP and PT. For each of the 4 companies excluding Brisa, daily data from 3-Nov-1997 to 29-Oct-1999 were available. For Brisa, quotation in BVL only started in 25-Nov-1997, and so available data starts in 25-Nov-1997 and also ends in 29-Oct-1999. Each of the resulting 495 records (479 for Brisa) includes the day's date, the closing value of the stock exchange main index (BVL30), the number of shares traded, and the opening, maximum, minimum and closing values of the stock.

From each companies' base data we constructed 15 daily-based "technical indicators" to be used as features to mine. These features are functions of the base data variables and summarize relations extracted from the previous 10 days of base data. As an example, one of the features expresses the relation between the 10-day and 3-day weighted moving averages of daily "reference values" (average of maximum, minimum and closing prices). Some of these features are categorical, while the others have integer or real values. However, the data mining algorithm requires discrete values, so we converted the original values of the features to discrete integer values ranging from 1 to 5 – the categorical features resulting in unordered sets of these values, and the numerical features resulting in ordered sets. As an example, the described relation between the 10-day and 3-day moving averages results in an ordered-value feature that is discretized the following way:

If (0.96 > (MA(10-day)/MA(3-day))) then the feature value is 1;
If (0.99 > (MA(10-day)/MA(3-day)) ≥ 0.96) then the feature value is 2;
If (1.01 > (MA(10-day)/MA(3-day)) ≥ 0.99) then the feature value is 3;
If (1.04 > (MA(10-day)/MA(3-day)) ≥ 1.01) then the feature value is 4;
If ((MA(10-day)/MA(3-day)) ≥ 1.04) then the feature value is 5.

The developed features were then subjected to a selection process to reduce their number to 10. This limitation on the number of features is introduced to help to reduce overfitting problems due to the scarce number of examples available in relation to the “descriptive power” of the full set of features. To select the 10 features to retain we applied (over the learning examples) a combination of methods including (with a heavier weight) Hong’s feature selection method [8] and also (with reduced weights) a measure of correlation between the feature value and the result to predict, and the simple information gain of the feature.

The final format of each prepared example consists of 10 decision features with 5 discrete values (classified as ordered or unordered) and one binary result attribute. The result attribute indicates, for each example, if the described “reference value” of the stock raises or falls in the next trading day. The total number of examples available for each stock is 478 (462 for Brisa). This number is smaller than the number of days in the original data mainly because several of the first days must be used to construct some of the features of the first example.

4 The PA3 rule induction algorithm

The rule induction algorithm we used, called PA3, is a recent general-purpose sequential cover algorithm [1]. The main features of PA3 include:

- A rule evaluation function that integrates explicit evaluations for rule accuracy, coverage and simplicity
- A rule generalization step that is run immediately after each rule is developed in an initial general-to-specific development phase
- A last rule filtering step that allows a choice of the tradeoff level between the accuracy and the global coverage of the final rule list.

The rule evaluation function is

$$v = a^{\beta} \times c^{1-\beta} + \chi s \ ,$$

where v is the rule value, a is the rule accuracy over the learning examples, c is the rule coverage, s is the rule simplicity and β and χ are constants that must be chosen according to the learning data characteristics (β regulates the relative importance of rule coverage and rule accuracy and χ regulates the importance of rule simplicity).

This evaluation function is used to direct the search and to choose among alternative rules during the initial general-to-specific rule development and also, in the following rule generalization step, to evaluate and choose possible generalizations of the rules that result from the initial general-to-specific development. In this generalization step the evaluation function of the standard PA3 is used with the same parameter values used in the general-to-specific rule development. This way, the algorithm only replaces a rule previously found by a more general version of that same rule if the latter is better according to the same evaluation measure.

PA3 induces an ordered list of "if…then…" rules. Each rule has the form "if <complex> then predict <class>", where <complex> is a conjunct of feature tests, the "selectors". In PA3, each selector implies testing a feature to see if its value is included in a specified range of values. So, each selector indicates the feature to be tested and the (inclusive) upper and lower limits of the range of values it has to be tested against. The postcondition of a PA3 rule is a single Boolean value that specifies the class that rule predicts for the cases that comply with all the selectors. It should be noted that, while a single PA3 rule includes a simple conjunction of tests, the final rule set is equivalent to a DNF formula.

PA3's last step uses a simple rule evaluation metric (different from the one used in the rule learning process) to filter the complete list of the induced rules, retaining only a reduced number of stronger rules. Since the rules learned by this algorithm form an ordered list, this rule filtering has to retain a set of the first contiguous rules (also maintaining the order of those rules). This filtering process is controlled by a user-defined parameter that must be set between 0 (to accept all the discovered rules) and close to 1 (to accept only the first, stronger, rules). Globally, this rule filtering method allows the user to choose the tradeoff level between a more complete case-space coverage and a reduced coverage using only the stronger rules (and therefore with greater accuracy).

5 Domain knowledge inclusion in PA3

Our global KDD process allows the integration and testing of domain theories of the "technical analysis" kind through a very simple process: They can be represented by the features generated from the original data. With this in mind, the theories that seem more useful when integrated at the rule induction algorithm level are "meta-theories" that can be globally applicable to the rules (in fact, combinations of "technical indicators") created by the rule induction algorithm from the data features. Since, in our domain, the relevant information present in the base data is almost completely "drowned" in noise, and overfitting tends to occur, we felt that the "meta-theories" to test should preferably be chosen to reduce overfitting.

One of the two theories we decided to test biases the learner against the selection of rules belonging to a particular class, while the other intends to promote rule generalization for another class of "marginal" rules. More specifically, the first theory states that a good rule should not include a test over an ordered-value feature that only accepts its middle value (3, since the range of possible values is 1 to 5), since that kind of "neutral" value for an ordered-value feature probably does not point strongly to clear changes in the stock value. To integrate this theory in the PA3 rule induction algorithm, we altered the evaluation of the basic (still unexpanded) rules: When, during the rule induction procedure, a rule has a selector involving an ordered-value feature with a value of 3, the standard evaluation result for that rule is multiplied by a constant (named *mod1*) with a positive real value smaller than 1, thus reducing the rule evaluation result. The second theory states that if a rule includes a test over a feature that has ordered values, and a value of 2 or 4 is accepted for that feature, then the corresponding "extreme value" (1 or 5 respectively) should also be accepted. The reasoning is that if a "strong" (high or low) value for a technical indicator seems to be predictive for the future behavior of a stock, then an even stronger (in the same direction) value for the that indicator should, most of the time, also point to the same prediction. To integrate this theory in the PA3 algorithm, we altered the evaluation of the rule expansions: When, during the expansion procedure, a rule has a selector (involving an ordered-value feature) that is expanded from a value of 2 or 4 to include (respectively) the extreme values of 1 or 5, the standard evaluation result is increased through multiplication by constant (named *mod2*) with a real value greater than 1.

The general idea behind this use of uncertain DK at the rule induction level is that if the theories are globally true, then the rules that do not agree with them have a greater probability of corresponding to statistic fluctuations found in the learning data, and not to stable patterns useful for out of sample prediction. This problem is originated by the noisy data and small learning set sizes and by the very large domain space searched. Introducing a small handicap in the evaluations of key rule classes ensures that the rules belonging to these classes that are present in the final rule list must correspond to patterns in the learning data with above-average "strength". Of course, if the theories are globally true, they should increase the out-of-sample accuracy of the predictions. If they are globally wrong, the out-of-sample predictions should present a reduced accuracy.

It is clear that increasing the number of learning examples reduces the advantages of integrating this kind of DK to focus the search, since, with a greater number of learning examples, the real patterns in the training data tend to be less obscured by noise. A marginal point to notice is that this biasing of the search will, of course, always reduce accuracy over the learning data.

6 Tests

Testing the integration of the domain theories over the available examples is not straightforward, because some characteristics of the domain and of the data limit the direct use of normal bootstrap or resampling methods.

In fact, the time series we intend to predict are far from deterministic, and their behavior can be expected to change over time due to changes in the underlying domain mechanics. This way, a prediction model that proves accurate during a certain time span can be expected to (progressively or suddenly) loose prediction accuracy in the future. This means that maintaining the temporal order of the examples is important if each test example prediction is expected to represent the real prediction setup at the time of that example. (As an example, consider the use of training examples immediately posterior to the test example being predicted: That corresponds to the use of context information that could not be available if the prediction of that example was required in a realistic situation, and can be expected to adjust the prediction model to the near-future domain behavior, artificially increasing the prediction accuracy).

This way, since the examples are "time stamped" and the domain behavior is expected to vary over time, we opted for the standard time-sequenced division of the examples, instead of a classic bootstrap or resampling method. To ensure unbiased test results, the available examples were divided into separate learning/validation and test sets. We used the first 300 examples (284 for Brisa) from each stock for learning and parameter selection and kept apart the last 178 examples from each set for testing.

To determine the best values for *mod1* and *mod2*, the first 200 of the 300 examples (184 of 284 for Brisa) were used for learning with different values for *mod1* and *mod2*, and the resulting rule sets were tested on the remaining 100 examples from the learning sets. The test results were averaged over the five stocks, and the best global values for *mod1* and *mod2* were selected.

Those values were then used to develop rule lists from the complete sets of 300 learning examples (284 for Brisa), and the prediction accuracy of those rule lists (over the test sets of 178 examples) was compared with the one achieved by rule lists obtained using the standard, unbiased, PA3 (*mod1* = *mod2* = 1).

PA3 uses 3 internal parameters:

- β and χ are used to regulate rule evaluation during the general-to-specific and specific-to-general rule development phases, and must be set considering the domain characteristics
- The final rule filtering parameter must be chosen according to the users desired tradeoff level between prediction precision and model coverage.

Since our aim with these tests is not to achieve the best possible prediction results, but to compare the results with and without the integration of the domain theories, we chose to simplify our test procedure setting, from the start, the β and χ parameters to the "standard" values of 0.8 and 0.01 [1], instead of optimizing them through tests over the training/validation data. Also to simplify the test procedure, the final rule filtering parameter was set to prevent any rule filtering, and a default rule was added to the end of each learned (ordered) rule set. This way, every learned model is guaranteed to produce a prediction for every possible test case.

During the initial test phase, to determine the best values for the two theories parameters, 7 values were tried for the *mod1* parameter (0.4, 0.5, 0.6, …,0.9, 1.0) and 11 values were tried for *mod2* (1.0, 1.1, 1.2, …, 1.9, 2.0). The results for each *mod1* value were obtained as an average over the *mod2* values and vice-versa. In all, 11 runs of the induction algorithm (over each of the 5 examples sets) are averaged to obtain each of the accuracy values for *mod1* and 7 runs (also over each of the 5 examples sets) are done for each of the accuracy values for *mod2*.

This test procedure does not try to optimize the *mod1* and *mod2* parameters for each stock. Naturally, each of the tested theories can present a different behavior over each stock involved in the study, and better final accuracy values could be expected if individual *mod1* and *mod2* values were used for each stock. However, considering the small number of examples available for each stock, we opted to use accuracy values averaged over the five sets, in order to obtain more robust values for *mod1* and *mod2*: This way, the chosen values are those that resulted in the global best results across the 5 stocks.

The average accuracy (as tested over the last 100 learning examples) of the rule sets learned over the first 200 (184 for Brisa) examples of each stock set is shown in percentage in Figure 1 for the tested values of *mod1* and *mod2*.

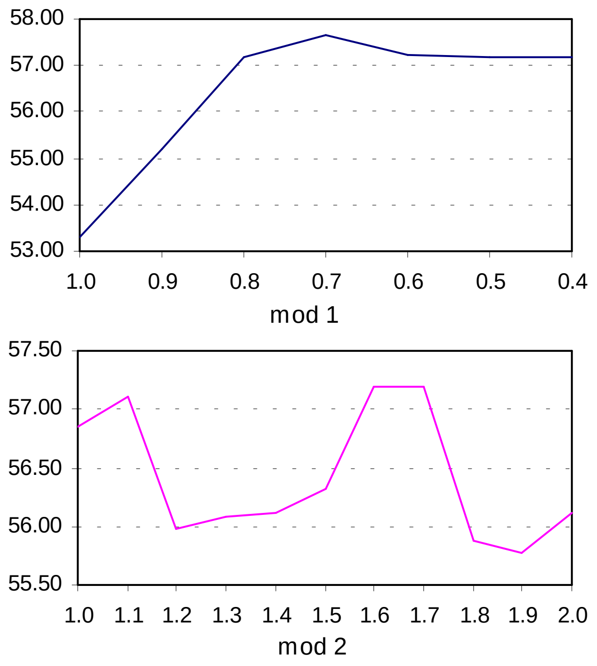

Fig. 1. Accuracy (in %) over the last 100 learning examples (averaged over the 5 stock sets)

As can be seen from the charts in Figure 1, for some of the tested modifier values both theories produce an improvement over the standard PA3 (*mod1*=*mod2*=1). However, the lack of regularity of the second theory chart contrasts with the "well behaved" first theory chart. In fact, in this first test, the second theory achieves an improvement for some of the *mod2* values, but several of the *mod2* values tested produce worse results than the basis value of 1.0. However, since these very simple first tests were based on relatively few examples and only intended to assist in choosing the values for *mod1* and *mod2* to be used in more extensive comparative tests, neither the stable behavior of the first theory nor the much less stable results for the second theory can be seem as very relevant.

Among the values tried for *mod1* and *mod2*, the best results were obtained for *mod1* = 0.7 and for *mod2* = 1.7. Those best values for *mod1* and *mod2* were then tested with rule sets developed over the sets of 300 learning examples (284 for Brisa), and applied over the five sets of 178 testing examples.

In these tests, a more complex procedure was used to try to achieve more stable results. Due to the method used to choose the "best" values for the two theory modifiers, and to the non-stationary nature of the time series involved, we wanted to keep a separation between the training and test sets, based on a strict time frontier: All the examples before that point are seen as training examples with a known outcome, and all the examples after that point are regarded as previously unseen test examples. That would lead to a simple holdout testing method that, due to the reduced number of available examples and to the small number of individual tests, would not produce reliable results, and would not allow a meaningful statistic significance analysis.

To try to circumvent this problem we opted for a test methodology that combines the simple holdout [10] and a modified bootstrap [5]. This test methodology uses 100 tests for each of the five stocks. In each of those tests, a model is learned on the basis of a bootstrap sample of the training examples (sampling examples from the original training set, using replacement, until a number of examples equal to the number in the original set is attained) and that model is tested over the complete, original, set of previously unseen 178 test examples. This way, each model is learned from approximately 63.2% of the training examples [5], and the models present exactly the same variability of standard bootstrap models learned over the training examples (in fact, they are learned exactly the same way). The tests, however, are always performed over the complete set of "out-of-sample" test examples (the best set of test examples we have), assuring that (unlike the standard bootstrap [10]) no optimistic "contamination" of results is possible. The bootstrap extraction of learning sets of examples is used only to generate variability, and results in a reduced prediction accuracy (because some of the training examples are left unused in the learning of each model) but maintains a fair test setting for the comparative tests of modifier values we want to conduct.

Table 1 shows the accuracy results obtained over the five data sets.

	mod1=1.0 mod2=1.0	mod1=0.7 mod2=1.0	mod1=1.0 mod2=1.7	mod1=0.7 mod2=1.7
BCP	55.88	55.81	55.99	56.82
Brisa	52.02	52.83	52.46	53.20
Cimpor	52.66	53.09	52.36	53.53
EDP	51.56	52.26	52.03	52.31
PT	57.19	56.95	57.97	58.15
Average	53.86	54.19	54.16	54.80

Table 1. Percentage accuracy for the neutral and best values of mod1 and mod2

Comparing the results of Table 1 (accuracy values close to 54%) and those indicated in Figure 1 (values close to 56%), a global accuracy decrease is clear. This decrease is mainly due to a very different behavior of the BVL stock exchange during the period corresponding to the learning examples (high volatility with a strong global raise) and during the period used to generate the test examples (a steady drop in the quote values). In those conditions, being able to achieve, over the test examples, global results clearly above the 50% level seems a strong indication that valid prediction patterns were in fact extracted from the training examples (both using the standard version of PA3 and using the versions with integrated DK). A secondary reason for the reduced accuracy is the test methodology that only uses about 63.2% of the available training examples to generate the prediction models, but this factor is partially offset by the increased number of available training examples in these tests.

The average results in Table 1 show that both theories, used in isolation, produce small accuracy improvements and that, used together, they result in a clearly greater improvement.

The results obtained for each stock show that when the integration of one of the theories in isolation results in a decreased accuracy (in only 3 out of 10 tests), the decrease if very small. The two theories combined always result in improved accuracy (in 5 out of 5 tests). This behavior seems to indicate that the average results can be regarded as relatively stable.

As previously referred, the prediction of this kind of financial time series would be impossible if the markets involved were theoretically efficient. That does not seem to be the case of most markets, and specifically of the market we are studying. However, even when stock markets are not theoretically efficient, that hypothesis does not seem to be very far from being true, and the predictability of stock quotes time series is always marginal. This way, in our binary prediction setting, a prediction accuracy close to 50% should be expected and any global percent accuracy improvement based in better data mining techniques must be marginal. This tends to result in a difficult setting for the analysis of the statistic significance of any data mining improvements. This global problem is compounded by the time-based sequential nature of the problems, by the relatively small number of available examples and by a large variability effect that can be associated with the noisy data [12].

To conduct a meaningful significance test of the theories integration, we used our bootstrap-based setup to analyze the number of times the altered algorithms achieved better, equal or worse results than the non-altered version (with mod1 = mod2 = 1.0). The involved test methodology allows us to conduct any desired number of tests with models that exhibit a bootstrap-like variance and still are tested in a strict holdout setup, allowing the tightening of the confidence intervals [13].

Table 2 shows the test results for the 100 runs for each of the five stocks that also produced the accuracy results of Table 1.

	mod1=0.7 mod2=1.0			mod1=1.0 mod2=1.7			mod1=0.7 mod2=1.7		
	B	E	W	B	E	W	B	E	W
BCP	50	6	44	49	5	46	54	8	38
Brisa	52	5	43	47	9	44	56	1	43
Cimpor	53	6	41	43	6	51	58	5	37
EDP	51	5	44	51	4	45	49	5	46
PT	47	4	49	52	3	45	54	4	42
Average	50.6	5.2	44.2	48.4	5.4	46.2	54.2	4.6	41.2

Table 2. Number of better, equal and worse results in relation to the basic, unmodified algorithm

One of the points that can be noticed in the results shown in Table 2 is the relatively large number of equal results. This is basically due to the fact that, in some runs, the mined data (that, in these tests, includes a number of repeated examples) is stable enough to generate exactly the same rule sets, in spite of the introduced search bias. Another interesting point is that in these results, only 2 of the 10 tests that compare the isolated theories with the unmodified algorithm produce more worse than better results (the slightly worse result of the isolated first theory in the accuracy results of the BCP stock is now inverted). This (average) worse accuracy result (see Table 1) is due to a small number of very bad results in some of the 100 accuracy tests (results that can be considered outliers).

The global results for each theory and for the two theories combined are consistent with the accuracy results shown in Table 1: In isolation, both theories produce a small but clear improvement over the unmodified algorithm, and the first theory produces a greater improvement then the second. When used together, the two theories produce a considerably greater improvement.

Applying a traditional significance analysis (single-sided paired *t* tests [13]) to the results in Table 2, the same general effects are detected: The average results for the first theory prove to be better than those of the unmodified algorithm version with 95% significance. The average results for the second theory are better than those of the unmodified algorithm version, but only with 68% significance. The average results for the two theories combined are better than those of the unmodified algorithm version with 98% significance. This last result seems to correspond to a meaningful prediction improvement in the difficult domain involved.

It should be pointed out that the test methodology we used expands the number of available examples by using simultaneous data from different stocks instead of a longer time frame of the same stock. This is common in stock time series data mining, but implies that the examples from each stock are not correlated which, of course, is not entirely true.

7 Conclusions

The work described in this paper reinforced our belief that direct use of DK in the core data mining phase of a KDD process can improve the overall efficiency of some knowledge discovery processes. In particular, changing the rule evaluation in order to introduce domain specific deformations in what would otherwise be an unbiased setting seems a promising way of integrating domain specific knowledge in the data mining phase of KDD processes that use rule induction algorithms.

As further work, we intend to test the present theories over more extensive stock market data. We also intend to evaluate, over the same domain, other globally applicable theories in the line of those involved in the present tests.

Acknowledgements

The author would like to thank L. Torgo and the anonymous reviewers of the paper submission for PAKDD-2001 for the helpful comments and corrections suggested.

This work was partially supported by the Portuguese Ministry of Science and Technology (MCT), through the Foundation for Science and Technology (FCT).

References

1. Almeida, P. and Bento, C.: Sequential Cover Rule Induction with PA3. To appear in Proceedings of the 10th International Conference on Computing and Information (ICCI'2000), Kuwait. Springer-Verlag (2000)
2. Choey, M. and Weigend, A.: Nonlinear Trading Models Through Sharpe Ratio Maximization. In Decision Technologies for Financial Engineering: Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets (NNCM-96). World Scientific (1997)
3. Cohen, W.: Compiling Prior Knowledge Into an Explicit Bias. In Proceedings of the Ninth International Conference on Machine Learning. Morgan Kaufmann (1992)
4. Cook, D., Holder, L. and Djoko S.: Scalable Discovery of Informative Structural Concepts Using Domain Knowledge. IEEE Expert/Intelligent Systems & Their Applications, 11 (5) (1996)
5. Efron, B. and Tibshirani, R.: An Introduction to the Bootstrap. Chapman & Hall (1993)
6. Fama, E.: Efficient Capital Markets: A Review of Theory and Empirical Work. Journal of Finance, May (1970)
7. Herbst, A.: Analyzing and Forecasting Futures Prices. John Wiley & Sons (1992)
8. Hong S.: Use of Contextual Information for Feature Ranking and Discretization. IEEE Transactions on Knowledge and Data Engineering, 9 (5) (1997)
9. Hutchinson, J.: A Radial Basis Function Approach to Financial Time Series Analysis. Ph.D. Dissertation, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (1994)
10. Kohavi, R.: A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95). Morgan Kaufmann (1995)
11. Lawrence, S., Tsoi, A. and Giles C.: Noisy Time Series Prediction using Symbolic Representation and Recurrent Neural Network Grammatical Inference. Technical Report UMIACS-TR-96-27 and CS-TR-3625, Institute for Advanced Computer Studies, University of Maryland, College Park, MD (1996)
12. LeBaron, B. and Weigend, A.: A Bootstrap Evaluation of the Effect of Data Splitting on Financial Time Series. IEEE Transactions on Neural Networks, 9 (1) (1998)
13. Mitchell, T.: Machine Learning. McGraw-Hill (1997)
14. O'Sullivan, J.: Integrating Initialization Bias and Search Bias in Neural Network Learning. Unpublished research paper from April 1996, available in: http://www.cs.cmu.edu/~josullvn/research.html
15. Pazzani, M.: When Prior Knowledge Hinders Learning. In Proceedings of the AAAI Workshop on Constraining Learning with Prior Knowledge. San Jose, CA (1992)
16. Weigend, A., Abu-Mostafa and Refenes A.-P. (eds): Decision Technologies for Financial Engineering (Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets, NNCM-96). World Scientific (1997)