International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:5, 2008

Applications of Genetic Programming in Data Mining

Saleh Mesbah Elkaffas, and Ahmed A. Toony

***Abstract*—**This paper details the application of a genetic programming framework for induction of useful classification rules from a database of income statements, balance sheets, and cash flow statements for North American public companies. Potentially interesting classification rules are discovered. Anomalies in the discovery process merit further investigation of the application of genetic programming to the dataset for the problem domain.

***Keywords*—**Genetic programming, data mining classification rule.

I. Introduction

FUNDAMENTAL analysis involves the analysis of economic data, industry conditions, company fundamentals, and corporate financial statements [1]. Data mining consists of the extraction of interesting novel knowledge from real-world databases [2]. Near boundless effort is expended in analyzing time series consisting of market and company metrics to predict future outcomes in order to achieve above average returns. This paper details an application of genetic programming to the problem of obtaining interesting (valuable) knowledge from the COMPUSTAT database (North America) [3] consisting of annual time series of income statement, balance sheet, and cash flow statement data for North American companies from the period 1972 – 1999.

References [4] and [5] propose a genetic algorithm [6] and [7] propose genetic programming frameworks respectively for induction of both classification and generalized rules from databases. The framework outlines a method for classification using tuple set descriptors (TSD) in the form of `WHERE` clauses for SQL queries and a count of rows of a goal attribute class matching the TSD. A sample TSD is of the form `((A1 >V1) AND (A2 < V2)) OR (A3 < A4)`, and represents a SQL query of the form in Fig. 1.

```
SELECT <goal attribute>, COUNT (*)
FROM <data table>
WHERE <tuple set descriptor>
GROUP BY <goal attribute>
```

Fig. 1 TSD and Goal Attribute form in SQL

Manuscript received September 26,2006.

S. M. Elkaffas, Department of Information Technology, Institute of Graduate Studies and Research, Alexandria University, Alexandria, Egypt (phone: +20 -10-5228715; e-mail: saleh.mesbah@gmail.com).

A. A. Toony, Department of Information Technology, Institute of Graduate Studies and Research, Alexandria University (phone: 0123334656; e-mail: ahmed_a_eltoony@yahoo.com).

The encoding of the sample TSD into prefix ordering for genetic programming manipulation is detailed in Fig. 2.

```
(OR
(AND
(> A1 V1)
(< A2 V2)
)
(< A3 A4)
)
```

Fig. 2 Encoding of Sample TSD for Genetic Programming Manipulation

A sample instance of a classification rule for a given goal attributes value is of the form:

```
IF <tuple set descriptor> THEN <goal
attribute> <relational operator> <class
designation value>.
```

II. Methods

All data mining tasks involve at least three steps: data preparation, data analysis, and decision-making. This work consists of the first two steps.

A. Data Preparation

The Standard & Poors COMPUSTAT database [3] contains 334 attributes, spanning 50 years of data for nearly 10,000 active (trading on public markets) and 11,000 inactive (non-trading, acquired, or failed) companies.

Some attributes for a given company at a given time may be unavailable (NULL); for instance, the database attributes are sparse for periods before 1972. In addition, the database is sparse for the most recent period (2001) for this edition of the database due to differences in fiscal year definition for different companies vs. the date of COMPUSTAT publication. For this reason, the period mined is 1972 – 1999 (a bug un-addressed prior to Genetic Programming runs left out 2000).

There are 78,263 rows for the 8,259 active companies during the period 1972 – 1999. In order to allow for cross-validation, a learning set of 4,264 companies and 39,054 rows was selected over the period and the remaining set of 4,265 companies and 39,209 rows were reserved for validation of the induced classification rules.

The data set is normalized by converting to a time series of period-to-period percent change of each attribute. In addition, n order to facilitate classification, the attribute value period-to-period percent changes are rendered to a discrete domain by rounding each percent change to the nearest discrete quantum expressed as a multiple of 5%. Percent changes in attribute

values of greater than 100% or less than –100% are arbitrarily mapped to +100% and –100% respectively.

Of the 334 available attributes, 52 are selected for mining based on two criteria: subjective estimation of the relevance of the attributes to the domain-specific classification problem, and a sparseness rule eliminating those attributes for which, during the 1972 – 1999 periods, more than 50% of the rows containing those attribute values were unavailable (NULL). The selected attributes are detailed in Table I.

TABLE I
COMPUSTAT ATTRIBUTES (METRICS) SELECTED FOR MINING

metric	%Change in …	metric	%Change in …
m01	Cash and Short-Term Investments	m27	Discontinued Operations
m02	Receivables	m28	Receivables-Estimated Doubtful
m03	Inventories	m29	Accounts Payable
m04	Assets	m30	Deferred Taxes
m05	Property, Plant, Equip	m31	Common Stock
m06	Long-Term Debt	m32	Treasury Stock Dollar Amt
m07	Sales	m33	Sale of Property, Plant, Equip
m08	Depreciation and Amortization	m34	Sale of Common and Pref. Stock
m09	Interest Expense	m35	Sale of Investments
m10	Special Items	m36	Purchase of Common and Pref. Stock
m11	Dividends-Preferred	m37	Receivables-Trade
m12	Dividends-Common	m38	Deferred Charges
m13	Price-CalYear-Close	m39	Accrued Expenses
m14	Common Shares Outstanding	m40	Prepaid Expense
m15	Employees	m41	Net Income (Loss)
m16	Intangibles	m42	Liabilities
m17	Debt in Current Liabilities	m43	Selling, General, Admin. Expenses
m18	Retained Earnings	m44	Extraordinary Items
m19	Invested Capital	m45	Short-Term Investments
m20	Cost of Goods Sold	m46	Receivables-Current-Other
m21	Advertising Expense	m47	Goodwill
m22	Research and Development Expense	m48	Notes Payable
m23	Rental Expense	m49	Capital Surplus
m24	Nonoperating Income (Expense)	m50	Stockholders Equity
m25	Interest Income	m51	Acquisition-Income Contribution
m26	Amortization of Intangibles	m52	Acquisition-Sales Contribution

B. Data Analysis

The technique used for data analysis is the aforementioned genetic programming framework for evolving TDSs for a given goal attribute. The goal attribute selected is change in stock price greater than 15% in an ensuing year. A single goal attribute and value was selected due to the experienced increase in evaluation time when attempting to find rules for each value of the goal attribute. The relation PRICE >= +15% was selected for its value as knowledge, should a consistent correlative classification rule be found.

The members of the terminal set for the TSD are of four types: the metric change in percent, the average of the percent change in a metric over a 1-3 year period, based on availability of data in the preceding periods, and including the current period for a given time t, and min and max over the same form of preceding and current time period. In addition 41 terminals are defined for the discrete percent change domain, e.g. –100%, -95%, …, 0%, 5%, … 100%. This results in a terminal set with 249 members.

The function set consists of the Boolean operators `AND, OR, NOT`, and the relational operators less than (<) and greater than (>). To deal with unavailability of a metric for a given company at a given time, all of the operators return `NULL` if any of their arguments is `NULL`. If the result of a TSD tree is `NULL`, rather than `TRUE` or `FALSE`, then during evaluation of fitness the TSD tree neither gains nor loses fitness as a result of processing the TSD for that row.

Initial and mutated TSD trees of depth N are constrained so that at depths 1 – (N-2), the only allowed functions are AND, OR, and NOT, and the only functions allowed at depth (N-1) are > and <. The crossover operation is limited to choosing crossover points that are internal nodes, i.e. functions. At depth N, the left-hand side of the operators > and < are constrained to terminals referencing the metrics or aggregates of the metrics. The mutation operator is allowed to choose any arbitrary point, with the constraint that if a terminal point is chosen, mutation can only introduce a new randomly selected terminal.

These constraints ensure that trees formed by the genetic operators remain valid as TSDs and are interpretable as IF...THEN classification rules. A sample TSD tree and its translation into intelligible terms are given in Fig. 3.

```
(or (not (or (and (not (> min(m21) min(m51)))
(> max(m52) max(m32)))
(or (> m46 avg(m26))
(> max(m19) -5%))))
(not (> min(m13) -65%)))
```

translated to

```
(or (not (or (and (not (> min(Advertising Expense)
min(Acquisition-Income
Contribution)))
(> max(Acquisition-Sales Contribution) max(Treasury
Stock Dollar Amt)))
```

```
(or (> Receivables-Current-Other avg(Amortization of
Intangibles))
(> max(Invested Capital) -5%))))
(not (> min(Price-CalYear-Close) -65%)))
```

Fig. 3 Sample TSD

The search space is approximately $\sum_{d=1}^{max\_depth} 2^{d-1} \cdot 249^{2^{d-1}}$ possible TSD trees, where 249 is the number of terminals, and 2^{d-1} is the number of internal nodes in a TSD of depth d. The "curse of dimensionality" is evident in the cardinality of the set of possible TSD's. Genetic programming runs evolving TSD trees were limited by memory size to a maximum depth of 30 and a maximum number of nodes of 3,000, thus failing to provide for full exploration of the TSD space, but leaving the hope that a valuable rule would be found in the diminished TSD search space.

For each row, the TSD tree is evaluated, possibly referring to 1-2 prior rows in the case of aggregates, and the goal attribute, price change >= 15% in the ensuing year, is determined. Using the two Boolean results, a count of true positives, true negatives, false positives, and false negatives is determined for all rows in the learning set. The fitness measure is the correlation of the TSD predicting the correct goal attribute across the entire learning set. As a specific case, trees of 1 node are given punishing fitness to work them out of the population, as they do not constitute valid TSD's. In summary, a Genetic Programming tableau is presented in Table II.

III. Results

The proposed classification was used for all genetic programming runs. Runs were executed with population size 5,000 on a 600 MHz-PC. The amount of computation cycles for fitness evaluation inhibited practical duration runs to ~10 generations.

The best of the generation 0 populations was found to have a correlation on the validation set proportional to the size of the population. The best of run was found invariantly in generation 1 for each population size. The additional generations resulted only in raising the mean fitness of the population towards the best of run correlation, while the mean TSD tree size also increased. Interestingly, the population included both large and small trees that achieved similar correlations. The highest correlation on the validation set for population 20,000 runs was ~0.08, and for the population 5,000 runs was ~0.05. Interestingly, the fitness for the best of generation rule was observed to decrease in some generations, but was recovered in subsequent generations.

TABLE II
Tableau for Classification

Objective:	Induce a classification rule using fundamental company metrics at year Y to predict >= +15% stock value appreciation at year Y+1
Terminal Set:	MDD, avg(mDD), min(mDD), and max(mDD), where DD in { 01,02,…,52}, and where aggregate functions avg(), min(), and max() have an implicit prior/current period range of <= 3 years.
Function Set:	AND, OR, NOT, <, >
Fitness Cases:	34,790 rows of attributes containing metrics at year Y and goal attribute (price) at year Y+1, where Y is in {1972, …, 1999}
Raw fitness:	Correlation: $C = \frac{tp.tn - fn.fp}{\sqrt{(tn+fn).(tn+fp).(tp+fn).(tp+fp)}}$
Standardized fitness:	C=(1-C)/2
Hits:	Not applicable
Parameters:	Populations: M=5000, 20000, 50000 Generations: G <= 50 $P_{crossover}$ = 0.85, $P_{mutation}$ = 0.05, $P_{reproduction}$ = 0.05
Success predicate:	C = 1.0 or max(C) within G generations

An example best of run individual TSD, with correlation 0.083, predicting 6,421 true positives, 13,212 true negatives, 8,486 false positives, and 7,164 false negatives is given in Fig. 4.

```
(and (not (or (and (> min(m09) 65%)
(< min(m51) min(m48)))
(and (< max(m45) max(m28))
(> min(m18) max(m05)))))
(not (not (not (< max(m26) avg(m13))))))
```

translated to

```
IF (and (not (or (and (> min("Interest Expense")
65%)
(< min("Acquisition-Income Contribution") min("Notes
Payable")))
(and (< max("Short-Term Investments")
max("Receivables-Estimated Doubtful"))
(> min("Retained Earnings") max("Property, Plant,
Equip")))))
```

```
(not (not (not (< max("Amortization of Intangibles")
avg("Price-CalYear-Close"))))))
THEN (Price > 15%)
```

Fig. 4 Best of Run Individual (correlation 0.083 on validation set)

It is interesting to note that the best of run individuals had a distinct propensity to prefer aggregate operators on the percent change metrics over the metrics themselves.

Suspecting that the "curse of dimensionality" was evident, subsequent runs were performed with population size 5,000 on the single processor Pentium-4, and 10,000 on the multiprocessor Itanium using only four metrics: Sales, Stock Price, Net Income, and Stockholder's Equity, performed against reduced learning and validation sets, specifically, the learning set was selected to be only performance metrics for Microsoft Corporation, and the validation set was selected to be only the performance metrics for Oracle Corporation.

For population 5,000 runs at G=100, classification rules with correlation 0.358 were discovered. These were significantly more correlative than the lesser generation classification rules discovered for the larger number of attributes and data sets.

Satisfyingly, mean population and best individual of generation fitness were observed to rise proportional to the number of generations. Interestingly, the best of generation 100 individual rule on the learning set produced a correlation of –0.0159 on the validation set. The second best individual had a correlation of 0.073 on the validation set, however, the third through tenth best individuals had correlations of 0.358. For population size 10,000 at G=16 (limited by time), the sixth best individual (on the learning set) achieved a correlation of 0.524, predicting 5 true positives, 9 true negatives, 1 false positive, and 1 false negative. Its form is detailed in Fig. 5, and it can be observed that the increase in tree size is proportional to prediction accuracy.

```
(and (not (not (not (not (or (not (< Price-CalYear-
Close 35%))
(or (< avg( Dividends-Common ) -45%) (< Stockholders
Equity max( Sales ))))))))
(and (and (or (or (or (and (> Net Income (Loss)
Price-CalYear-Close )
(< avg( Sales ) 10%))
(and (< min( Stockholders Equity ) 25%)
(< max( Net Income (Loss) ) avg( Net Income
(Loss)))))
(or (and (< Price-CalYear-Close min( Stockholders
Equity))
(> min( Net Income (Loss) ) -95%))
(and (< avg( Stockholders Equity ) -50%)
(< max( Net Income (Loss) ) -95%))))
(or (not (or (< max( Sales ) Sales )
(< max( Sales ) 15%)))
(not (or (< max( Stockholders Equity ) -40%)
(> Price-CalYear-Close min( Sales ))))))
(and (not (not (or (> min( Net Income (Loss) ) -85%)
(< Net Income (Loss) -80%))))
(and (or (and (< min( Stockholders Equity ) 100%)
(> max( Sales ) Net Income (Loss) ))
(not (> max( Price-CalYear-Close ) 70%)))
(or (not (> avg( Dividends-Common ) 5%))
(and (> max( Stockholders Equity ) -85%)
(> Net Income (Loss) max( Stockholders Equity)))))))
(and (or (or (or (not (> avg( Net Income (Loss) )
max( Sales )))
(and (< max( Net Income (Loss) ) 50%)
(> avg( Net Income (Loss) ) 5%)))
(or (not (> Price-CalYear-Close max( Sales )))
(not (> max( Price-CalYear-Close ) -90%))))
(and (not (not (> avg( Net Income (Loss) ) avg(
Dividends-Common))))
(or (and (< avg( Dividends-Common ) min( Net Income
(Loss)))
(> max( Stockholders Equity ) 5%))
(or (> Net Income (Loss) max( Stockholders Equity ))
(< min( Price-CalYear-Close ) Net Income (Loss))))))
(not (not (or (not (> avg( Sales ) avg( Sales )))
(not (> min( Net Income (Loss) ) avg( Sales
))))))))
```

Fig. 5 Classification Rule (6 best of run, M=50,000) With Highest Correlation (0.524) On Validation Set

It is also interesting to note that the best of run individuals with the highest validation set (Oracle) correlation based on a learning set (Microsoft) correctly classified both Oracle's stock drop in 1991, and it's rise in 1999, based on data from 1990 and 1998 respectively, processed by the discovered classification rules. Additional work is required to determine the significance of such results.

IV. Discussion

Certainly the underlying large data set is noisy. The mystery of the early plateau in fitness and the early generation discovery of apparent maximum correlation, with both large numbers of TSD attributes and large datasets, seem to imply a local maximum discovered in the search space, or a pragmatic failure to process a large enough population for enough generations to better explore the search space. One area investigated was the impact of NULL TSD results over the learning set. This was found to not be an operative factor in the evaluation of fitness in any of the runs. It is interesting that some rules with positive correlation are discovered, classifying the validation set more correctly than not correctly.

Further work is required to determine if the discovered classification rules confer an equity trading advantage over common trading strategies such as buy-and-hold, or exceeding the performance of the S&P 500 index. Owing to the results to date and the low positive correlations in classification rules discovered for large datasets and large numbers of attributes, further investigation is required before the classification rules were to be put to real-world use, if at all. The results on the smaller learning and validation sets for much smaller number of relevant attributes is promising, but cannot be absolutely determined to be because of the metrics: both companies were in an industry known to grow significantly in net wealth creation over the concerned time period.

Additionally, other considerations, such as which the terminals are defined such as the lack of looking at other companies in the same industry or observation of key economic data and market indices limits the view of the TSD's. Ratios commonly used in fundamental analysis may or may not be useful to add to the set of metrics available to the TSD's. No attempt was made to use Genetic Programming to rediscover these or novel other relevant ratios. It is also not certain that any form of fundamental analysis is significantly predictive of future equity prices.

V. Conclusion

A framework for using genetic programming to induce classification rules has been applied. Classification rules with positive correlation for predicting price changes greater than or equal to 15% were discovered, with prediction on the validation set being more right than wrong. It is difficult to determine if better rules could be discovered in the TSD search space, due to practical limits computational cycles and memory capacity for evaluating significant populations of TSD trees over reasonably large data sets comprising reasonably large numbers of metrics, over a significant number of companies and time series histories. Further work is required to investigate the reasons for the plateau of fitness across generations of genetic programming runs for larger numbers of attributes and larger datasets.

References

[1] Little, J. B. and Rhodes, L., "Understanding Wall Street", Liberty Publishing Company, Cockeysville, MD, 1983.
[2] Fayyad, U.M., Piattetsky-Shapiro, G., and Smyth, P., "From data mining to knowledge discovery: an overview. Advances in Knowledge Discovery and Data Mining", 1-34. AAAI/MIT Press, 1996.
[3] COMPUSTAT, http://www.compustat.com/www/db/na_descr.html, Standard & Poors Institutional Market Services, Englewood, CO, USA, 2001.
[4] Fidelis, M.V., Lopes, H.S., and Freitas, A.A., "Discovering comprehensible classification rules with a genetic algorithm", Proc. Congress on Evolutionary Computation - 2000 (CEC-2000), 805-810. La Jolla, CA, USA, 2000.
[5] Freitas, A.A., "A genetic programming framework for two data mining tasks: classification and generalized rule induction", Genetic Programming 1997: Proc. 2nd Annual Conf. (Stanford University, July 1997), 96-101. Morgan Kaufmann, 1997.
[6] Goldberg, David E., "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison Wesley Longman, Inc, 1989.
[7] Koza, John R., "Genetic Programming: On the Programming of Computers by Means of Natural Selection", Cambridge, MA: The MIT Press, 1992.