

遺伝的プログラミングによるルール生成と債券格付分析への応用

01304556 九州大学 *時永 祥三 TOKINAGA Shozo

02756076 信州大学 池田 欽一 IKEDA Yoshikazu

1 まえがき

債券格付の方法としてニューラルネットワークや判別分析などがあるが、言語的な記述ではないため、実際の解釈が不便である。本報告では、遺伝的プログラミング (Genetic Programming: GP) により格付のルールを生成する方法を提案し、数値実験を行なう。

2 プロダクションルールによる格付

アナリストが実施する格付を自動化する数値的な手法として、ニューラルネットワークや判別分析が提案されている。一方、プロダクションルールを用いる方法では、変数の相互関係が論理式で記述されるので、解釈が容易である利点がある。この分野ではエントロピー法に基づく Inductive Learning やその拡張形があるが、最適性が保証されない。本報告では、GP によるルール生成を用いる。

ルールは条件部 C と実行部 P で if C then P のように記述され、例を次に示す。

```
if x1>0.2 and (x3*x4 >1.2) then class='A'
```

ルールは算術式および比較演算子からなる命題と、その論理演算により構成される。

GP においては、これらのルールを木構造 (prefix representation) で表現し、これらを個体とよぶ。個体を複数用意しておいて、格付を的確に実施するルールに対応する個体に大きな数値 (適合度) を与える。GP においては、適合度の高い個体を選択し、交差処理などの遺伝的操作を加え、その結果として判別を向上させる。

説明のため GP による関数近似について述べる。最初に関数を木構造 (個体とよぶ) で表現することとし、1つの個体が1つの関数近似に対応する。木構造の表現として、演算記号と演算子を適切な順序に1列にならべる表現である前置表現 (prefix representation) を用いる。個体の関数値を計算するには、変数に値を与えて解釈する。

個体の初期値および2つの個体を部分的に切断して交差処理により相互に接続する処理の結果として生成される個体が、関数を表現しているかどうかをチェックする目的のため、*StackCount* という変数を導入する。*StackCount* は、スタックの中に存在する演算子の数から、演算記号の数を引いた数値である。従って、前置表現において、その途中まで、この *StackCount* をかぞえて、その数値が同じであれば、この位置で2つの個体を切断し交差処理をしても意味のある方程式表現を与えていることになる。

GP を適用するアルゴリズムは以下ようになる。

(ステップ 1): 個体の初期値生成

最初の個体の集合 (プール) を乱数をもとにして発生させる。この場合、すでに述べた *StackCount* を用いて、関数として意味をなすものが得られるまで繰り返す。

(ステップ 2): 個体の適応度の計算

すでに述べた関数近似における予測値と観測値との2乗誤差の逆数を、個体 i の適合度 S_i として定義する。個体を適応度の大きい順に並びかえておく。

(ステップ 3): 交叉処理

適応度に応じて2つの個体を取り出し、交叉処理を実施する。交叉処理では、乱数を1個発生させておいて、一方の個体の切断箇所とし、ここまでの *StackCount* の値を計算する。こののち、もう一方の個体の *StackCount* を計算しながら、同じ *StackCount* の値となる場所を検出し、これらから任意に1個を選択して切断位置を決める。

(ステップ 4): 突然変異

GP における突然変異として、グローバル突然変異ではある個体 (木構造) I_s を生成し、目的とする木構造の葉の部分 I_s により置き換えるローカル突然変異は木構造の葉の部分 I_s を、別の変数に置き換える操作である。

3 論理式に対する GP

論理式も木構造で表現され、非ターミナルノードには論理演算子は配置され、ターミナルノードには命題が配置される。更に、これらの命題は、すでに述べた予測モデルにおける算術式と同様に、木構造で表現される。

以下では、個体を表現するデータ構造を、やや簡単化するため、論理式は2項論理演算に分解してされていると仮定する。すなわち、論理演算を記述する場合に、その論理演算の対象とする被演算子は2つに限定する。すなわち、複雑な論理演算は、被演算子2項からなる論理式で記述される。

この場合、個体表現には階層構造を用いることが便利であり、ルールの最上位を示すルールの全体構造、論理演算の表現を第1階層の個体(これを、レベル1の個体プールとよぶ)としておき、論理演算の対象となる命題は別の個体として表現し、プール(これを、レベル2の個体プールとよぶ)を構成しておく。この間の対応関係をリンクとして保存しておく。

個体が与えるものは、論理演算の結果として求まる論理値である。

(1) 算術式の値の計算

最初に、レベル2の個体である算術式を解釈する。学習サンプルとした与えられたデータには、変数の値が示されているので、これを変数に代入する。

(2) 命題の論理値の解釈

算術式の値をレベル1の個体に代入したあとで、それぞれの命題の値を計算する。比較演算子と、これが対象とする算術式があり、算術式の値はすでに分かっているので、命題の論理値が求まる。

(3) レベル1の個体の解釈

それぞれの命題の論理値が計算されるので、個体の取りうる最終的な結果が計算される。

4 遺伝的操作について

算術式における GP において、演算子、被演算子を、それぞれ、論理演算子、命題により置き換えることにより、まったく同様の操作が可能となることが分かる。前のケースでは、演算子、被演算子を数えあげる目的で用いた *StackCount* を、ここでは、論理演算子、命題を数えあげるために用いる。ただし、交差点が命題になっている場合には、半分の確率で、命題の内部での交差処理を行う。

2つの個体における交差点が命題になっている場合には半分の確率で、対応する2つの命題(これらはレベル2の算術式の個体である)に対して交差処理を行う。

個体との交差処理を実施するグローバル突然変異と、論理式の任意の位置の記号を、同じ種類であるが異なる記号で置き換えるローカル突然変異がある。

5 応用例

2000年度における89社の決算データ(26財務指標へ集約)と格付データを用いて、本報告の手法による格付の性能評価を行なう。すなわち、外的基準として格付の値(A, B, Cの3つに集約)を与え、これに一致するように GP を用いて学習をすすめ、最後に同じ学習データに対して求められた最も適合度の高いルールを適用する。その結果を外的基準と比較して、一致度を求める。

比較のため、同様に解析をニューラルネットワーク(NN)と判別分析による方法についても実施しておく。これらの結果を表1にまとめている。これより分かるように、本報告に示した手法は、他の手法と同等か、やや優れた性能を与えている。

表1. 格付分析の結果(識別率%)

	ランク A	ランク B	ランク C
本手法	78.1	86.6	79.1
判別分析	76.7	84.3	77.8
NN	78.2	86.6	79.2

6 むすび

今後、更に業種や企業の数を増やした場合の解析、言語的な解釈を与えることや、この簡単化を進める予定である。

参考文献

- [1] J.R.Koza, Genetic Programming, MIT Press, 1992
- [2] Y. Ikeda and S.Tokinaga, "Approximation of chaotic dynamics by using smaller number of data based upon the genetic programming", Trans. IEICE, vol.E83-A, no.8, pp.1599-1607, 2000
- [3] Y. Ikeda and S.Tokinaga, "Controlling the chaotic dynamics by using approximated system equations obtained by the genetic programming Trans. IEICE, vol.E84-A, no.9, pp.2118-2127, 2001.
- [4] 矢加部 正幸, 時永祥 三, "遺伝的プログラミングを用いた CNN による拡散モデルの近似と同期化への応用", 電子情報通信学会論文誌, vol.E85-A, no.5, pp.548-559, 2002.