"Discovery" algorithm and existence theorem.

"DISCOVERY" TOOL expresses patterns in first order logic and assigns probabilities to rules generated by composing patterns. As any technique based on logic rules, this technique allows one to obtain human-readable forecasting rules that are interpretable in financial language and it provides a forecast. An expert can evaluate the correctness of the forecast as well as forecasting rules. "DISCOVERY" TOOL and related software systems [Vityaev E. 1983; Vityaev, Moskvitin, 1993] generalize data through "law-like" logical probabilistic rules.

Conceptually, law-like rules came from the philosophy of science. These rules attempt to mathematically capture the essential features of scientific laws: (4) high level of generalization, (2) simplicity (Occam's razor), and (3) refutability. Formally, an IF-THEN rule C is

$$A_1\& \ldots \& A_k \Rightarrow A_0,$$

where the IF-part, $A_1\&...\&A_k$, consists of true/false logical statements $A_1,\ldots,A_k$, and the THEN-part consists of a single logical statement A_0. Statements A_i are some given refutable statements or their negations, which are also refutable. Sub-rules can be generated by truncating the IF-part, e.g.,

$$A_1\&A_2 \Rightarrow A_0 \text{ , } A_1\&A_2\&A_3 \Rightarrow A_0.$$

For rule C its conditional probability

$$Prob(C) = Prob(A_0/A_1\&...\&A_k)$$

is defined. Similarly conditional probabilities

$$Prob(A_0/A_{i1}\&...\&A_{ih})$$

are defined for sub-rules C_i of the form

$$A_{i1}\& \ldots \& A_{ih} \Rightarrow A_0, \text{ where } \{A_{i1}, \ldots, A_{ih}\} \subset \{A_1, \ldots, A_k\}$$

Conditional probability $Prob(C) = Prob(A_0/A_1\&...\&A_k)$ is used to estimate the forecasting power of the rule to predict A_0. The rule is "law-like" iff all of its sub-rules have less conditional probability than the rule, and the statistical significance of it is established. Each sub-rule C_i generalizes rule C, i.e., potentially C_i is true for a larger set of instances [Mitchell, 1997]. Another definition of "law-like" rules can be stated in terms of generalization. The rule is a "law-like" rule iff it cannot be generalized without producing a statistically significant reduction in its conditional probability. "Law-like" rules defined in this way hold all three properties of scientific laws. They are: (1) general rules from a logical perspective, (2) simple, and (3) refutable. Chapter 5 presents some rules extracted using this approach.

"DISCOVERY" TOOL and software [Vityaev E. 1983; Vityaev, Moskvitin, 1993; Vityaev 1992] searches all chains

C_1 , C_2 , …, C_{m-1}, C_m

of nested "law-like" subrules, where C_1 is a sub-rule of rule C_2 , C_1 = sub(C_2), C_2 is a sub-rule of rule C_3, C_2 = sub(C_3) and finally C_{m-1} is a sub-rule of rule C_m, C_{m-1} = sub(C_m). Also,

$$Prob(C_1) < Prob(C_2), \dots , Prob(C_{m-1}) < Prob(C_m).$$

There is a theorem [Vityaev, 1992] that all rules, which have a maximum value of conditional probability, can be found at the end of such chains. The algorithm stops generating new rules when they become too complex (i.e., statistically insignificant for the data) even if the rules are highly accurate on training data. The Fisher statistical test is used in this algorithm for testing statistical significance. The obvious other stop criterion is the time limitation.

Below "DISCOVERY" TOOL is presented more formally using standard first-order logic terms described in Section 4.4.3. Let us consider the set RL of all possible rules of the form:

$$A_1\&\dots\&A_n \Rightarrow A_0 \qquad (4)$$

where $A_0,A_1,\dots,A_n$ are atomic formulas in one of the following forms
$P^e(t_1,\dots,t_k)$ and $(t=g)^e$,

$$t(z^t_1,\dots,z^t_{lt}),\ g(z^g_1,\dots,z^g_{lg}),\ t_1(z^1_1,\dots,z^1_{l1}),\ \dots ,t_k(z^k_{k1},\dots,z^k_{lk})$$

are terms,

$$z^t_1,\dots,z^t_{lt},\ z^g_1,\dots,z^g_{lg},\ z^1_1,\dots,z^1_{l1},\ \dots ,\ z^k_{k1},\ \dots ,z^k_{lk}$$

are individual constants, and e = 1(0) is an indicator if there is a negation of the predicate. If e=1 then there is no negation of a predicate and if e=0 then a predicate is negated. For instance, $(t=g)^0$ means that $t \neq g$, and similarly,

$$P^0(t_1,\dots,t_k) \Leftrightarrow \neg P(t_1,\dots,t_k).$$

The formula (4) means that for every substitution of instances for the individual constants, if the IF-part is true then the THEN-part will also be true. In this substitution, objects from data D represent individual constants.

The goal of "DISCOVERY" TOOL is finding BST rules, defined below. BST stands for "best".

<u>Definition</u> 1. Rule $C = B_1,\dots,B_l \Rightarrow A$, where $l > 1$ and $\mu(B_1,\dots,B_l) > 0$

is called a BST rule for atomic formula A and data D if and only if:

1. $\mu(C) = \mu(A/B1\&\dots\&Bl) > \mu(A)$, where B1&…&Bl is a condition generated using data D,
2. Rule C has maximum of conditional probability $\mu(C)$ among rules satisfying condition 1 and generated by the same data, and
3. For any rule C* satisfying conditions 1, 2, we have $C \Rightarrow C^*$.

This means that BST rule C is the strongest rule.

Condition 1 means that rule C has a reasonable If-part (premise), i.e., the conditional probability μ of rule C is greater then the probability of atomic formula A itself. If this condition is not satisfied then there is no reason to add the premise for forecasting A. If atom A has a high probability itself then it can be predicted without a premise.

Condition 2 brings the "strongest" rule, i.e., the rule with maximum conditional probability among rules satisfying condition 1 above for the same data.

Condition 3 means that a BST rule is the most "general" among rules satisfying conditions 1 and 2, i.e., a BST rule covers the widest set of cases from D for which it can be applied. Formally, this idea is presented in Definition 2.

Definition 2. Relation $C \rightarrow C'$ is true for rules C and C' if

$$\{A_1,\ldots,A_n\} \subset \{A'_1,\ldots,A'_{n'}\},$$

where $C = A_1\&\ldots\&A_n \Rightarrow A_0$ and $C' = A'_1\&\ldots\&A'_{n'} \Rightarrow A_0$, $n,n' > 0$.

Now the task is to find the set of all BST rules for data D. It is well known that the exhaustive search of all rules (4) for finding the set of BST rules is practically impossible if predicates with more than one variable are used. Therefore, the search should be constrained to a lesser set of rules, but still allowing the discovery of BST rules. To accomplish this task use the following definition and theorem.

Definition 3. Rule (4) is called a regularity if it satisfies the following conditions:

$$\mu(A_1\&\ldots\&A_n) > 0,$$

$$\mu(A_0/A_1\&\ldots\&A_n) > \mu(A_0/\text{SubConjunction}(A_1\&\ldots\&A_n))$$

where μ is the probability of an expression, and

$$\text{SubConjunction}(A_1\&\ldots\&A_n) = A'_1\&\ldots\&A'_k,$$

$$\{A'_1,\ldots,A'_k\} \subset \{A_1,\ldots,A_n\},\ \{A'_1,\ldots,A'_k\} \neq \varnothing.$$

Theorem 1. If rule R is a BST rule for data D, then rule R is regularity on data D [Vityaev, 1992].

Let us denote the set of all regularities (rules) by RG. It follows from Theorem 1 that the task now is to find the set of rules RG. Having RG we will be able to make all "best predictions" using regularities from RG.

Partition the set of regularities RG to get chains, called chains of semantic probabilistic inference [Vityaev, 1992],

$$C_n \succ C_{n-1} \succ \ldots \succ C_2 \succ C_1 \succ A,$$

where $\succ$ is a symbol for semantic probabilistic inference.

Actually, the task consists of finding all semantic probabilistic inferences. The following heuristic rule is used as the base for a practical implementation of semantic probabilistic inference:

all regularities can be found by searching all chains beginning with a regularity with a short IF-part.

This heuristic is used in "DISCOVERY" TOOL for arranging the search of regularities. In spite of the use of a heuristic no one BST rule will be lost. "DISCOVERY" TOOL like FOIL and FOCL operates with Horn clauses (Section 4.4.3). However, the theory was developed for a general case (see the mentioned theorem [Vityaev, 1992, Vityaev et al, 1995]).
Below we use Horn clauses

$$A_1\&\ldots\&A_n \Rightarrow A_0 \qquad (5)$$

where $A_0,A_1,\ldots,A_n$ are atomic formulas (literals) of the form

$$P_t^e(z^t_1,\ldots,z^t_t),\ t = 0,1,\ldots,n.$$

Each P_t^e is a predicate and $z^t_1,\ldots,z^t_{lt}$, t = 0,1,…,n are individual constants (constants for short). Two formulas of type (5) are equivalent if each of them can be obtained from the other by a one-to-one replacement of constants. We will denote all non-equivalent formulas of type (5) as a RuleSet.

The concept of Data Type was defined in Section 4.9. Each data type is associated with a specific set of predicates. Actually, these predicates and their properties define a data type. Several examples of these predicates are presented in Section 4.10.2.

Let $Pr = \{P_{i\in I}\}$ be a set of predicates associated with a specific data type and let $Z=\{z_1, z_2,\ldots\}$ be a set of individual constants. Atomic formulas A, which are predicates from Pr or their negations defined on a set of constants from Z are used. Below are presented core concepts used in MMRD.

Regularity: This is a special type of formulas from the RuleSet. For instance, in a financial time series, constants $z^t_1,\ldots,z^t_{lt}$ can be represented by days with properties:

$$(z_1 <_{S\&P} z_2)\&(z_2 <_{S\&P} z_3)\&\ldots\&\ (z_{n-1} <_{S\&P} z_n) \rightarrow (z_n <_{S\&P} z_{forecast}),$$

where $z_1 <_{S\&P} z_2$ means that the SP500C is smaller on day z_1 than on day z_2 and that this rule forecasts $z_n <_{S\&P} z_{forecast}$, i.e., SP500C for the next day will be greater than SP500C on day z_n if the given IF-part of the rule is satisfied. More regularities for financial applications are presented in Chapter 5.

Next a formal concept of regularity type, allowing us to distinguish regularities from other formulas is also introduced. The general intuitive idea of a formal concept of regularity is that regularity represents the structure of relations between components. It is important to note that the regularity type is defined not as a property of an individual formula itself, but as relations of this formula with other formulas. In particular, a chain of formulas is introduced as a part of regularity type. These formulas are ordered by a "semantic probabilistic inference"(➢) relation like

$$C_n \succ \ldots C_i \ldots \succ C_2 \succ C_1 \succ A\ .$$

All rules C_i are regularities. The last rule C_n is called a BST regularity. This ordering already was called a semantic probabilistic inference, because $C_{i+1} \succ C_i$ means that probability of C_{i+1} is greater than the probability of C_i and there is a logical relation (sub-rule) between IF-parts of C_i and C_{i+1}

In this way, a heuristic is introduced into a search process, which makes the search in the hypothesis space tractable. The "DISCOVERY" TOOL begins from simplest rules and generates a tree of these chains implementing a kind of Branch-and-Bound method. This method begins from Breadth-First search among formulas of a limited length. This set of formulas is called an initial set of rules.

Suppose that initial rule C_n is obtained. This rule is a tail of some chain like

$$C_n \Rightarrow ....C_i ... \Rightarrow C_2 \Rightarrow C_1 \Rightarrow A .$$

It is possible, in the simplest case, to have just $C_n \Rightarrow A$. Then a set of possible additions $\{C_{k+1}^{Possible}\}$ for C_k are generated by calling a function Specilize_rule(C_k), which generates possible addtions

$$Specilize\_rule(C_k)=\{C_{k+1}^{Possible}\}.$$

Specialization using function Specilize_rule(C_k) is guided by the strength of attribute scales (expressed in data type). At first, "DISCOVERY" TOOL adds predicates expressing the simplest (rough) data types like nominal scale. Further properties of more complex scales like ordering and interval scales to refine a rule are added. We also need a Boolean function IsRegularity(R,D) which will test if R satisfies the definition of a regularity for data D. This function is true if R satisfies the definition and it is false if R does not satisfy the definition.