A Selective Induction Framework for Improving Prediction in Financial Markets

Sung Kun Kim*

Abstract

Financial markets are characterized by large numbers of complex and interacting factors which are ill-understood and frequently difficult to measure. Mathematical models developed in finance are precise formulations of theories of how these factors interact to produce the market value of financial asset. While these models are quite good at predicting these market values, because these forces and their interactions are not precisely understood, the model value nevertheless deviates to some extent from the observable market value. In this paper we propose a framework for augmenting the predictive capabilities of mathematical model with a learning component which is primed with an initial set of historical data and then adjusts its behavior after the event of prediction.

Keywords : Induction, Financial Markets, Option Pricing, Incremental Learning

---

Received : 2015. 06. 24.　　Final Acceptance : 2015. 09. 04.

※ I pay tribute to Prof. James Clifford of New York University who unexpectedly had passed away for an acute disease while working on this paper.

* Professor of Information Systems, School of Business, Chung-Ang University, Seoul, KOREA, e-mail : skim@cau.ac.kr

1. Introduction

Induction is an important aspect of human intelligence, enabling us to generalize behavior patterns from a collection of individual observations, and thereby to predict future behavior in similar circumstances [Al-Jarrah, 2015]. It is clearly a valuable tool in human learning as well as human problem solving. While it manifests itself in a variety of forms, it can be viewed as a process of partitioning a universe of objects into subsets or classes in such a way that each class is meaningful with respect to some purpose or goal. Such a partitioning is largely dependent upon the relevance of the initial object descriptions to the class description that is to be learned. Michalski [1983] characterizes induction methods on the basis of the degree of this relevance as below:

Complete Relevance All descriptors used in depicting objects axe directly relevant to an inductive assertion, which in most cases is a mathematical expression of a general form underlying these descriptors.

Partial Relevance Given a large number of descriptors, a significant portion of which are irrelevant, the learning task is to select the most relevant ones for the class description. This task is called selective induction.

Indirect Relevance Given an initial set of indirectly relevant descriptors, the learning task is to construct descriptors that become directly relevant to the class descriptions. This task is called constructive induction.

A number of recent learning systems are based on the second approach, selective induction. Examples include AQ1X [Michalski and Chilausky, 1980], Meta-Dendral [Feigenbaum and Buchanan, 1993], and CILA [Bloedorn and Wnek, 1995]. What is to be learned differs from system to system. For example, AQ11 discovers diagnosis rules for soybean diseases while Meta-Dendral discovers new cleavage rules for mass spectrometer.

Consider an example from among many possible problems in finance which could be approached by means of an inductive problem-solving strategy. Determining or predicting the market value of financial assets (e.g., stocks, bonds, options, etc.), a value which, is greatly affected by anticipated returns from the asset, is a complex, fuzzy decision-making process since the realities of financial markets do not allow for revealing accurately its anticipated returns. Such a decision can be supported by a formal asset valuation model, in that the model can determine the "intrinsic" value of asset (the asset's estimated worth in the market) and this value can be used as one yardstick for the decision. However, such model-estimated prices generally deviate, sometimes in structurally determined ways, from the market-determined prices. Characterizing these structural deviations from a collection of performance results of the model, with the goal of predicting the market value more accurately, is a classic example of an induction problem. One may view a description about the model's performance for a specific asset at a specific moment in time as an object, while a pattern of structural deviations derived from an analysis of these objects would be a class.

In general an inductive assertion about objects in the universe is called a hypothesis. Induction systems manipulate these hypotheses in such a way that objects within a class (objects described by the hypothesis) are similar or meaningful with respect to some goal. An important component in induction systems is a particular criterion in evaluating goodness of hypotheses according to sample data. Some induction systems employ a binary evaluation function. The range of values for the binary evaluation function is 0 or 1 (that is, negative or positive instance). Alternatively, this evaluation function can be defined more flexibly as the proportion of correct categorization of objects; the range of values is [0, 1]. This function which characterizes the degree of goal satisfaction of objects is called the utility function [Fayyad et al., 1996].

In selective induction descriptions of classes or hypotheses involve no descriptors other than those used in describing objects. In general, descriptors of objects can be chosen as abstract features or elementary primitives. For instance, one may represent a checker board using an abstract feature such as piece advantage, or using an elementary primitive such as the contents of the individual squares. One general requirement for selective induction systems is that these descriptors should be aggregates or abstract features that cause the utility function to be uniform, smooth, or locally invariant [Rendell, 1983].

In the case of a financial model, one kind of attribute in describing objects (instances of the pricing model's performance results) is domain-specific knowledge which explains the financial market's behavior. Financial domain knowledge, however, is generally characterized by incompleteness and fuzziness, and cannot be used as abstract features. When these instances are described in primitive domain knowledge, the inductive system will provide a very irregular utility function, making the induction task more complex. To avoid this problem, inductive systems in the financial domain require a scheme in which more abstract aggregates or generalizations are derived from elementary primitives.

The other requirement for selective induction systems in finance is that incremental learning is necessary, due both to incomplete domain knowledge and to noisy data. In other words, patterns of structural deviation (or classes) learned at one point should be modified to accommodate more data supplied subsequently, thereby averaging out over the time the influence of irrelevant or invalid noisy data.

The purpose of this paper is to describe a framework for selective induction systems in finance or other domains which are characterized by the unavailability of abstract features and the existence of extensive, noisy data. In Section 2 we review some incremental learning strategies from the literature on machine learning. Our proposed framework is described in Section 3, and some conclusions about this approach, are presented in Section 4.

2. A Review of Incremental Learning Approaches

One of the requirements for selective induction systems in a financial domain is the use of an incremental learning strategy; when more examples become available, hypotheses learned at one stage should be modified to accommodate them. While most selective induction systems are capable of accommodating more data supplied subsequently, they differ with respect to how experience from one stage is fed back into the induction process. We categorize the incremental learning strategies taken in existing selective induction systems into two kinds: the minimal approach and the iterative approach. These two approaches are described in more detail below.

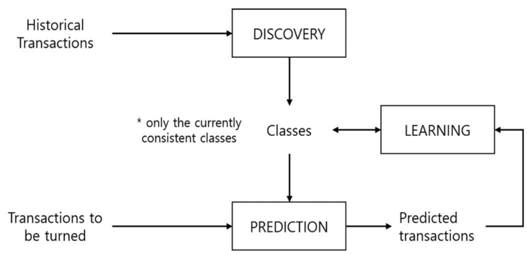
2.1 The Minimal Approach

In this approach when new training examples are available which are not consistent with the current hypothesis, this hypothesis is modified. A hypothesis in this approach is considered consistent if and only if it agrees with all positive examples and no negative examples.

Though systems employing this strategy seek to find a hypothesis that is consistent with all of the training instances obtained so far, the set of hypotheses which had been maintained previously does not have to be considered. An important assumption made by these systems (e.g., the version space [Mitchell, 1982] and AQ11 [Michalski and Chilausky, 1980]) is that the training examples contain no errors and that the generalization language is sufficient to describe the target concept.

For instance, the version space strategy attempts to maintain the set of hypotheses which are plausible at any given time by specifying two boundary sets which are of maximally specific generalizations or maximally general generalizations consistent with the observed training instances. These sets are denoted S and G, respectively. The version space initially starts out as the whole space of all possible hypotheses. As more training instances are obtained, some hypotheses are removed from this version space. More specifically, each positive instance makes the set S generalize and each negative instance makes the set G specialize. This process is continued until these two boundary sets are identical, or until no new examples are available. <Figure 1> illustrates the basic features of this approach.

The target concept is found when these two boundary sets become identical. At this point there is no further need of incremental learning; new training instances which completely agree with this concept will be determined as positive instances, and all other instances as negative. Thus, while the concept is incompletely learned, this strategy utilizes a minimal incremental learning approach because there is no need to look up either previous training instances or previously maintained hypotheses. Once the concept is completely learned, in the sense that the sets S and G are identical, no further incremental learning is employed.



<Figure 1> The Minimal Approach to Incremental Learning

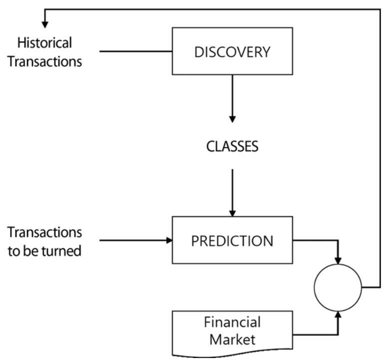
It is easy to see why this approach fails to acquire the correct concept when errors exist in training instances. Any noisy instance can make these two boundary sets, S and G, overly generalized or overly specialized. Once some hypotheses are eliminated from the version space, they cannot be recovered even though it may have been an erroneous instance which caused the elimination. For these reasons, this strategy is not a viable one for finance problems which abound with noisy data.

2.2 The Iterative Approach

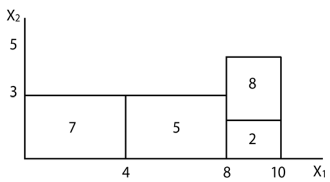
While the minimal approach revises the current hypotheses only when new training examples do not agree with them, the induction process in the iterative approach is rerun from scratch on all the available instance including the "new arrivals". The essence of the iterative approach is to partition massive amounts of data into maximally dissimilar classes. Each time the system is run to construct new classes, all the training instances supplied thus far will be partitioned. This approach is shown in <Figure 2>.

An example of systems employing this approach is the Probabilistic Learning System (PLS) [Rendell, 1983]. The domain of PLS is heuristic search problems. Each node developed in the search tree is mapped as a point in feature space. For example, a node in the search tree for the fifteen puzzle may be defined using features such as 'sum of distance of tiles from home' and 'number of pair reversals'. The goal of PLS is to construct a predictive heuristic function based on the feature vector. The probability of a node's being on a goal state, not the path distance from the goal state, is used as a basis for this heuristic function. This probability is called utility. The strategy employed is to partition a universe of objects (nodes) so that one class is meaningfully different from another while objects within the class are similar; that is, each class will have roughly similar utility.

In the fifteen puzzle problem each class is described rectangularly, or more specifically, as a conjunction of feature ranges, as shown in <Figure 3>. As an example, the leftmost rectangle in Figure 3 represents the class more formally described as $(0 < x1 < 4) \cap (0 < x2 < 3)$. It denotes that objects within this class (that is, nodes in the search tree which are described so in terms of the feature vector) will appear in a goal state with a probability of .7. The use of such rectangles as an economical way to represent and to manipulate classes has also been shared by other induction systems (e.g., [Samuel, 2000]).



〈Figure 2〉 The Iterative Approach to Incremental Learning



〈Figure 3〉 An Example of Class Representation

The induction task is to develop a set of classes such that objects within a class will become consistent or similar according to the utility function. More specifically, the task begins with one large region which covers all objects; nothing more specific has yet been learned. PLS splits a region into two subregions in the most desirable way. In an attempt to ease computational complexity which may be caused in finding the best hyperplane for splitting, PLS makes a decision that regions should remain rectangular and be oriented with the feature space axes. The inductive criterion used in determining the dissimilarity measure between two classes reflects the uncertainty around descriptions of classes. This inductive criterion will be discussed and compared in more detail in the next section.

This iterative strategy has the capability to accommodate noisy data and facilitates the process of incremental learning. At the conclusion of one iteration, the new experience gained is fed back into the set of objects used in constructing the original classes, and the class construction process is rerun from scratch. Though this approach allows for the discovery of maximally (or optimally) dissimilar classes, its computational requirements can be enormous. This problem is especially apparent in financial induction systems, which have at their disposal a huge amount of data (transactions) for analysis in the construction of classes.

2.3 Comments on the Above Classification

The two approaches discussed above presume that training examples are supplied to the learning system on an ongoing basis, not all at once. In other words, the training examples supplied for the construction of initial classes do not correspond to the complete set of training examples available in the domain.

In some situations, the learning system needs to discover a most efficient classification under the assumption that all available examples are given at the time of construction. In such cases, an incremental learning approach is unnecessary. Nevertheless, should that assumption prove false, the system could be easily converted to reflect new training examples through the adoption of an iterative incremental learning approach.

For instance, ID3 [Quinlan, 1983] seeks to create a decision tree which correctly classifies all the given objects. Here each object is described in terms of a fixed set of attributes, each having its own set of possible values. Initially, the decision tree has one root node which corresponds to the whole universe. Then the system selects one attribute whose information-theoretic value appears to be greater than the others, and constructs a set of child nodes based upon the values of the selected attribute. This process continues until no more discrimination is needed or the attributes are exhausted. As a result, the depth of the decision tree is at most the number of attributes. In this way the final decision tree becomes minimal in the sense that the expected number of tests to classify an object is minimized.

Suppose objects continue to be supplied to the classification system, and the system aims to repeatedly discover a most efficient classification procedure from the set of objects available at that time. Then it is obvious that whenever more objects are available the classification process can be rerun based on the whole set of examples including the "new arrivals." The decision trees which had been previously maintained have no influence on the one yet to be discovered; this is equivalent to the iterative incremental learning approach. Similarly, some learning systems [Hunt and Stone, 1966] requiring the existence of the complete set of training examples can be easily modified to do iterative incremental learning

3. A Contingent Incremental Learning Approach

The minimal incremental learning approach discussed above has limited use in financial applications, due both to the inability of this approach to handle noisy data, and to the incomplete state of the domain knowledge in many financial applications. The enormous computational requirements inherent in the iterative approach also contraindicate its use in financial domains which have a wealth of available data. In this section we present a variation on the incremental approach, which we call contingent incremental learning, which reconstructs classes or concepts only when they are no longer coherent. Since this approach maintains the set of constructed classes explicitly over time, the system can select for reconstruction of the class most susceptible to instability due to the addition of new examples. Whether to maintain or invalidate a class already "discovered" by the system will be contingent explicitly on its coherence. In order to be able to fully illustrate this approach, we first must describe the representation of objects and classes in the system. This will lead to our further discussion about a technique for deriving from primitive descriptors more aggregate ones.

3.1 Representation of Objects

In general, selective induction systems require that attributes used in describing objects be abstract features so that the utility function indicating the degree of goal-satisfaction of objects can be smooth. One kind of attribute for explaining the behavior of a financial market is domain-specific knowledge.

Domain knowledge in finance, however, is incomplete and fuzzy. For instance, a piece of domain knowledge is just one factor out of hundreds or perhaps thousands of factors explaining the deviation of model prices from actual prices. The exact consequence of that particular factor, however, cannot be easily expressed. Furthermore, the interdependence among many factors is complex and not well-understood. In this environment, therefore, an induction system would provide very irregular utility functions making the induction task more complex. To avoid this problem, induction systems in financial domains need scheme of deriving abstract aggregates from fuzzy factors.

One type of aggregate description which can be derived is the average performance results of the asset valuation model for a set of transactions which are identical with respect to a certain range of market influence. For example, one such aggregate factor might be the average deviation of the model's prices from the actual prices for options on the same underlying stock.

In finance it is common practice to distinguish different ranges of market influences; as an example, the financial market's influence on a particular asset can be separated into a number of different ranges of influence, or scopes. Scopes range from very general (e.g., all options traded on the same day) to more and more specific (e.g., all options with the same expiration date which are traded on the same day). <Table 1> describes four such scopes of influence for the case of stock options. As a particular example, we could look at some specific options and associate them with these scopes, as shown in <Figure 4>. So, for instance, one may say that two different options within the box labelled Scope 3 would be influenced identically with respect to Scope 3.

Thus, the number of features used in describing the pricing model's performance in a particular transaction viewed as an object will correspond to the number of these scopes. The value of each attribute is the average deviation of model prices from actual prices for all transactions which were described identically according to that scope. So we can represent an object as a vector $X = (x_1, x_2, \cdots, x_n)$ where n is the number of attributes or features and x1 corresponds to an average deviation of model prices from actual prices with respect to the i^{th} scope.

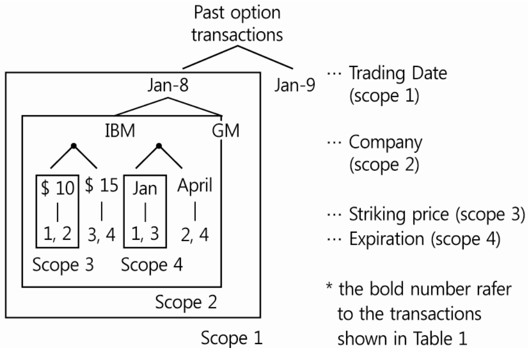
3.2 Representation of Class

We are using induction in the sense of partitioning a universe of objects-in this case the pricing of a particular object on a specific day - into subsets or classes in such a way that each class is meaningful with respect to some purpose or goal. The degree of goal-satisfaction of an object x is called its utility. One may interpret it as the credibility of the hypothesis "object x will contribute to the goal", where the goal here is to reduce the difference between the model-predicted price and the observed market price.

One way of measuring this utility is as the probability of the hypothesis that the object x will be in the desired class. In supervised learning where there is a priori classification of examples, or where there exists a teacher who can definitely determine examples as positive or negative, the utility function u(x) is 1 or 0. Otherwise, u(x) is some probability value between 1 and 0.

〈Table 1〉 Four Scopes of Influence on Options

Scope 1	*a range of influence where all options in the market on a particular day axe affected identically;*
Scope 2	*a range of influence where all options with the same underlying stock which are traded on the same day are affected identically;*
Scope 3	*a range of influence where all options with the same striking price which are traded on the same day are affected identically*
Scope 4	*a range of influence where all options with the same expiration date which are traded on the same day are affected identically*



No.	Trading date	stock	striking price	expiration
1	Jan. 8, 1987	IBM	$10	Jan.
2	Jan. 8, 1987	IBM	$10	April
3	Jan. 8, 1987	IBM	$15	Jan.
4	Jan. 8, 1987	IBM	$15	April

〈Figure 4〉 Options and Associated Scopes

In our financial model a class is represented by its description and behavior. First of all, a class is described by a number of features, each feature reflecting the average deviation of model prices for a scope. As in PLS, its description is a conjunction of feature ranges $(a_1 < s_1 < b_1) \cap \cdots \cap (a_4 < s_4 < b_4)$ where si refers to the ith scope. If a class were described with only two features, it could be represented by a rectangle as shown in <Figure 3>. For example, a class or concept represented by the leftmost rectangle in <Figure 2> is described as $(0 < x_1 < 4) \cap (0 < x_2 < 3)$. In the general case involving n features, an n-dimensional space is required.

Along with its description, we also represent the behavior of a class as the probabilities of the occurrence of some event. The event in PLS, for instance, is concerned with whether a particular state in the state space will lead to a goal state. The leftmost class in <Figure 2> represents that objects within this class will lead to the goal state at the probability of .7. In our financial model the event is the performance of an asset valuation model compared to the actual market prices. In other words, a class's behavior is concerned with whether the asset pricing model would overprice (or underprice) actual prices of the transactions belonging to this class. One may classify the deviation in the model's performance with an arbitrarily graduated scale. For some purposes a scale with seven gradations might be sufficient, such as "highly overpricing", "moderately overpricing", "slightly overpricing", "nearly accurate", "slightly underpricing", "moderately underpricing", and "highly underpricing." Thus, a class's behavior is represented in terms of a probability for each of these scales.

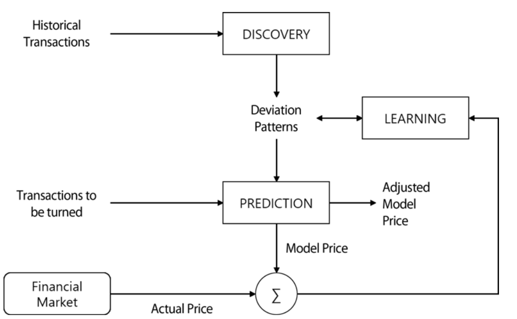
3.3 Discovery of Initial Classes

In general, the induction task is largely dependent on the kind of information supplied to the induction system. Financial data accumulate continuously and in an uncontrollable manner. Since capturing all of this continuous stream of data is usually impractical, the financial analysts typically use some periodically aggregated data (e.g., daily, weekly, monthly) in studying financial processes. Once generated, these data are stored as historical information.

This issue raises the following question: How much historical data (training examples) are needed in constructing an initial set of classes?. The comparatively low quality of financial data suggests that the induction system be offered a large amount of data for constructing the initial classes. Since vast amounts of historical data are always available, the system designer can employ an arbitrarily large period of historical data.

Until the initial classes are constructed, the induction task cannot be evaluated. After its construction, the system attempts to predict the behavior of the financial markets. In an incremental learning system, a comparison of the system's prediction with the actual prices will invoke reconstruction of these classes. Accordingly, a general financial induction system includes three subtasks: initial discovery, prediction, and reconstruction; <Figure 5> shows a general architecture of such a system.

Initially, the induction system begins with one large region (or polyhedron) in n- dimensions which covers all historical data of a certain initial period. This is the initial "class", which initially includes every transaction. Using the known performance results on historical data, the system then attempts to refine this class, which it does by splitting this region into two subregions in such a way that each subregion is meaningful with respect to the performance results of an asset pricing model in financial markets. Such splitting is continued until it is determined that each region cannot be further split into two subregions, each coherent in its own right. The main question involved in this process is how



⟨Figure 5⟩ A General Architecture of Financial Induction Systems

to select a hyperplane which would split one region into its two most dissimilar subregions.

The criterion for assessing hypotheses is called the inductive criterion [Watanabe, 1969] or inductive bias [Mitchell, 1980]. The study of methods to evaluate this inductive criterion has been one of main interests for machine learning research. Quinlan's ID3 [Quinlan, 1983] uses an information theoretic approach to discover a classification rule (decision tree) for a collection of objects, each being described in terms of a fixed set of attributes. The idea in this approach is to order these attributes according to their information-theoretic value; the information-theoretic value for each attribute is measured and the one leading to the largest increase in this value is selected as the one which contributes the next branch of the decision tree. In this way the final decision tree becomes minimal or the expected number of tests to classify an object is minimized.

As an alternative to ID3, Rendell [1983] proposed a probability-based discrimination measure. This is based on a dissimilarity measure which takes into account the probabilistic utility for two subclasses under consideration. One may derive from this dissimilarity measure the information-theoretic measure employed in ID3. Since this measure also reflects the statistically defined error factor, it tends to produce more reliable inductive criterion for small or non-random samples [Rendell, 1986]. However, whether two classes axe similar or dissimilar is dependent on a specific value of this measure. More specifically, if this measure gives a positive value, two clusters are evaluated as dissimilar, otherwise as similar. However, this measure may not be applied when the behavior of a class is scaled into more than these two gradations of positive and negative, (e.g., into underpricing, nearly accurate, and overpricing.)

Hart [1986] proposed the x2-statistic discrimination measure as an alternative to ID3 s information-theoretic approach. The ID3 algorithm selects one attribute with the largest increase in the information-theoretic value for the next branch of the tree. Since it does not discriminate between a substantial increase and a negligible increase, the resulting decision tree can be highly sensitive to small changes in the training set. However, the x^2-statistic measure selects only the attribute with the highest and sufficiently significant x^2-value. Moreover, the level of significance can be easily adjusted by using various critical values for this x^2 statistic.

Consider attribute A with possible values A1, ⋯, Aα and classes C_1,⋯, C_γ. Let the total number of cases considered be N. Then, the x2-value is defined as shown in <Figure 6>. Here oi,j is the number of observed cases with value Aj in class C_i and is the expected number of cases with value Ai in class Ci. The idea behind the use of the x^2-statistic is that a particular attribute will not be a good discriminator when the values of the attribute are randomly distributed over these classes. In other words, any clumping or deviation from randomness will signify a good discriminator. So the question is to choose the best discriminator, that is, the attribute with the highest x^2-value.

However, one attribute with the highest x^2_ value may turn out to be insignificant at a given critical level. One useful characteristic of the x^2-statistic is that the significance of the x^2-value can be set to a certain critical value (e.g., 5%). Whenever the highest x^2-value attribute is determined as insignificant, the splitting process will halt.

The algorithm shown in <Figure 7> refines the region h, generating two refined regions called h_1 and h_2. The use of the threshold-critical-value, and the fact that the choices on where to split a region are based upon discrete increments in the range of each regions, guarantees that only a finite number of potential hyperplanes is considered, and that eventually further splitting will not produce a sufficient x^2-value; therefore the algorithm eventually will terminate, yielding a finite set of regions.

$$X^2 = \frac{\Sigma(e_{i,j} - o_{i,j})^2}{e_{i.j}}$$

where

$o_{i,j}$ = the number of cases with values A_j in class C_i

$e_{i,j} = \dfrac{\Sigma_k o_{i,k} \Sigma_l o_{l,j}}{N}$

df = (i−1)×(j−1); degrees of freedom

〈Figure 6〉 Definition of x^2-statistic

3.4 Reconstruction of Classes

The initial classes learned at one point should be modified to accommodate more data supplied in the future. However, the iterative incremental learning approach may not be practical due to its enormous computational requirements. To keep the reconstruction cost at a minimum, the system has to be able to identify those classes susceptible for reconstruction. This can be done by maintaining explicitly the previously constructed classes. This idea is similar to the version space [Mitchell, 1982] in that both strategies take a conservative approach to modifying classes (or concepts). However, the version space method keeps only a set of currently consistent hypotheses, not the previously maintained ones, thus making it impossible to refute decisions made at earlier stages.

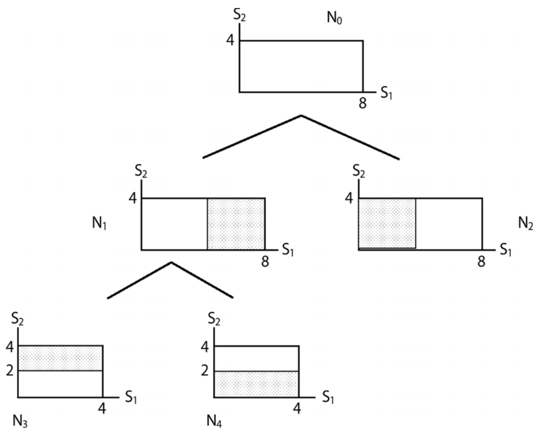
The initial classes formulated by the clustering algorithm in Section 3.3 will form a binary tree structure. The root node in the tree refers to the database of all objects and the clustering algorithm splits a node into its two descendant nodes. <Figure 8> shows an example of such a tree. The root node of each subtree has non-empty left and right son nodes.

After its construction of initial classes, the induction system can perform the desired goal of predicting the behavior of the financial markets. When predicting, only the leaf nodes of the tree will be employed since they represent the current set of most-specific classes. In other words, the prediction process is concerned with which of the leaf classes will be applied to predict an asset pricing model's behavior for a given transaction. This prediction will be performed for each of transactions which are supplied to the induction system at any time.

At the conclusion of a round of predictions, the predicted objects will be attached to the corresponding leaf classes. In order to maintain the set of classes coherently over time, the degree of cohesiveness of these leaf classes should be determined again. The same x^2-statistic can be used for this.

SPLIT(h)
  best-hp, best-value, subregion 1, subregion 2 := 0
   WHILE ∃ any untried hyperplane, oriented with the axes,
  DO
     Let a hyperplane chosen be *current-hp*
     Let the two temporary subregions of h be h_1 and h_2
     Let the x^2-value of *current-hp* be *Measure*
     If *Measure* is significant at level of α (say, 5%)
       Measure > *best-value*
     THEN
       best-hp := *current-hp*
       best-value := *Measure*
       subregion 1 := h_1
       subregion 2 := h_2
  IF *best-hp* ≠ *0*
THEN
       H1 := *subregion 1*
       H2 := *subregion 2*
       SPLIT*(H1)*
       SPLIT*(H2)*
ELSE
    Quit

〈Figure 7〉 The Clustering Algorithm



〈Figure 8〉 Example of the Class Tree

Recall that when discovering these initial classes, a certain critical value was used as a threshold to determine the coherence of classes. If the x^2-value was below the level of this threshold, the class was treated as incoherent, otherwise, as coherent. One would use the same critical value for determining the coherence of the leaf-classes after more objects were added to them. Whenever a leaf-class produces a x^2-value below the threshold value, its parent sub-tree would be reevaluated for possible splitting or further invalidating. This approach may lead to frequent invalidation of classes, deteriorating the efficiency of the induction system due to continual computation.

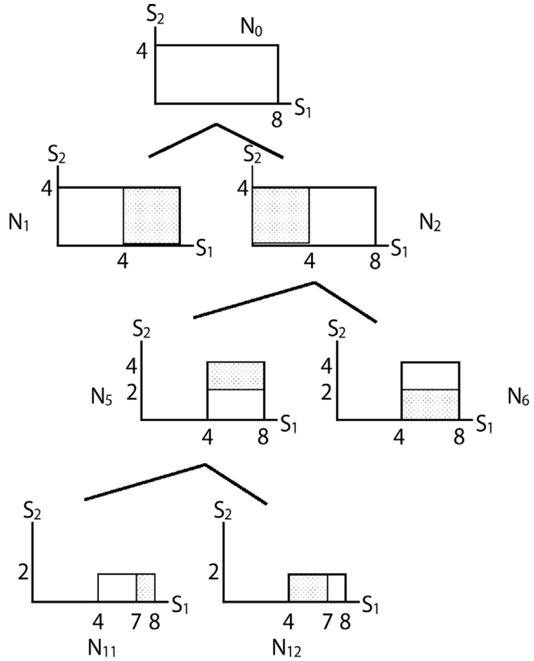
One approach effective in reducing the computational requirements of the refinement process is to use two threshold values, an upper and a lower. The upper threshold will be used for determining whether classes are coherent, and the lower one for determining whether they are incoherent. Thus, a leaf-class will be invalidated only when its x^2-value is insignificant at the level of the lower threshold. We call this strategy contingent incremental learning.

<Figure 9> shows the contingent learning algorithm. Those leaf nodes which are "candidates" for reevaluation because they have more training instances attached, have their significance recomputed by this algorithm. Depending upon the value of this significance, a leaf class may be further split or invalidated. If the probability is below the lower threshold, the leaf-class is invalidated. In this case, its parent will become a leaf-node and all of the descendants of this new leaf node will be invalidated.

```
RECONSTRUCTION
Let a set of the attached leaf-nodes be L
WHILE L≠ 0
DO
Select from Lone leaf-node, l, with the lowest depth
L := L-{l}
    BEST-DISCRIMINATOR(l)
    IF best-value < the lowest threshold
          THEN Let the father node of l be f
                L := L+{f }-{f's descendants}
    ELSE
          IF best-value > the highest threshold
          THEN
                F1 := subregion 1
                F2 := subregion 2
                SPLIT(F1)
                SPLIT(F2)
          ELSE
            quit
```

〈Figure 9〉 The Reconstruction Algorithm

<Figure 10> shows this reconstruction process. Suppose that two leaf nodes, n2 and n3 in the original tree were used for prediction and were candidates for reevaluation. Moreover suppose that 5% and 10% are used for the upper and lower threshold values, respectively. If the leaf-node class is significant at the level of the upper threshold, it is further split as in the initial splitting process. So, for example node n2 is split. Since node n3 is insignificant at the level of the lower threshold, node n3 is made a leaf node, invalidating all of its descendants. If a leaf class is significant at the level of the lower threshold, but not at the upper one, no modification of this node is made.



〈Figure 10〉 An Example of Class Refinement

We believe that this two-valued threshold approach may be natural to problems in finance, since a financial expert's risk-taking behavior in the financial markets can usually be described using such a range. Such range-determined behavior motivates our use of an upper and lower threshold on the cohesiveness of a class. Since different experts may have different ranges, the contingent learning strategy can to this extent be individualized to take into account a particular financial expert's own range of risk-taking behavior.

4. Evaluation of Our Approach

The main objective of our approach is to predict option prices more accurately than the Black-Scholes option pricing model by adjusting the behavior of the Black-Scholes option pricing model. We call our approach designed to tune an option pricing model as TOP. The first hypothesis is described as below:

H1: The ratio of deviations of TOP's prediction and the actual prices is less than the ratio of the option pricing model's prediction and the actual price.

It was also our belief that TOP would predict more accurately with the set of deviation patterns whose cohesiveness have been maintained over time than with the initial set of deviation patterns which may no longer be cohesive. The second hypothesis is described as below:

H2: The ratio of deviations of TOP's adjusted model price from actual price is smaller than that of the 'unlearned' adjusted model price of TOP.

4.1 The Evaluation Approach

Constructing a set of initial deviation patterns needs a considerable amount of historical data. For our evaluation, we used three months of data (from 5/16/84 to 8/15/84) of past daily option transaction data. The performance evaluation of TOP that we conducted was based on the following month of past option transactions (from 8/16 to 9/15/84). An adjusted model price for each of these upcoming options was compared against its actual price with an assumption that the actual price is unknown at the time of TOP's prediction.

The option prices data was obtained from IDC. Historical volatilities for these underlying stocks were gathered Daily Graph for the corresponding period. For the computation of the risk-free rate, the current market price of a U.S. Treasury bill maturing at about the same time as the option was taken from Wall Street Journal. Other data used in describing the model's behavior were gathered from many different sources such as Standard and Poor's Daily Stock Record, Federal Reserve Bulletin, and Daily Graph.

4.2 Evaluation Results

A paired, one-tailed t-test was performed to show the result as below:

Ratio of Deviation	# of cases	Mean	Std. Deviation
TOP	389	12.46	11.74
B-S model	389	15.06	19.44

t-value: -2.75, 1-tail Probability: 0.003.

Since the sign of t is the opposite of that expected, our hypothesis is unsupported. A careful analysis was undertaken to re-examine our evaluation scheme. Option price, in general, is directly related to the divergence between its striking price and the current price of its underlying stock, since the holder of an option can make a profit only when the striking price is less than the current price of the underlying stock on the expiration date. We therefore decided to examine the TOP's performance in these two different cases (in-the-money and out-of-the-money options), since it was suspected that the TOP's poor performance was related to a different variability of these two groups of options.

The same t-test was performed separately for these two groups. The results are as below:

Group 1: In-the-money option

Ratio of Deviation	# of cases	Mean	Std. Deviation
B-S model	200	8.42	7.05
TOP	200	6.82	5.32

t-value: -3.74, 1-tail Probability: 0.000.

Group 2: Out-of-the-money option

Ratio of Deviation	# of cases	Mean	Std. Deviation
B-S model	187	16.87	14.02
TOP	187	23.96	24.59

t-value: -3.83, 1-tail Probability: 0.000.

What we can conclude form the above analysis is that, for the group of in-the-money options, TOP performed better than the option pricing model while the option pricing model by itself worked better for the group of out-of-the money options. As a matter of fact, out-of-the-money options are not under the consideration for investment because investors could purchase the real stock for a lower price.

We also tested the second hypothesis to produce the following results:

Ratio of Deviation	# of cases	Mean	Std. Deviation
Unlearned	389	19.21	26.71
Learned	389	15.06	19.44

t-value: 4.82, 1-tail Probability: 0.000.

Since the sign of t is as expected and the one-tail probability is smaller than the critical value, our hypothesis that TOP with the 'learning' capability performs better than the 'unlearning' TOP is supported.

5. Conclusions

This paper provides a framework for selective induction systems in those financial domains characterized by the unavailability of abstract features and by the existence of extensive historical data which are nevertheless likely to be noisy. To derive abstract features which will be used in describing objects, the induction system groups into a class a set of objects characterized identically according to a certain range of market influence. Toward this end, one may separate the influence of the financial market on a given asset into any number of different ranges.

The proposed induction framework utilizes a contingent incremental learning approach which adjusts its classification scheme only when the classes are deemed to be no longer coherent. This approach explicitly maintains a set of previously constructed classes organized as a binary tree. After an initialization phase based upon a set of historical data, the system enters a dual mode of a phase of predictive activity followed by a phase of contingent learning. The performance results of a predictive phase are used in the subsequent learning phase to readjust the current set of coherent classes, The coherence of each affected class is reevaluated, potentially leading to further splitting or invalidating. The use of two threshold values to guide this phase prevents excessive invalidation of classes which might be caused by noisy data. In this way, the induction system can maintain a coherent set of classes over time. The statistical measure of coherence appears meaningful since it seems to model the risk-taking behavior of financial expertise.

The implementation of the prototype and its subsequent evaluation provide tangible evidence that this approach is viable.

References

[1] Al-Jarrah, O. Y., Yoo, P. D., Muhaidat, S, Karagiannidis, G. K., and Tahaam, K., "Efficient Machine Learning for Big Data: A Review", *Big Data Research*, Vol. 2, No. 3, 2015, pp. 87-93.

[2] Buchanan, B. and Feigenbaum, E., "Dendral and Meta-Dendral", *Artificial Intelligence*, Vol. 11, 1978.

[3] Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P., "Knowledge Discovery and Data Mining: Towards a Unifying Framework", *KDD*, Vol. 96, 1996.

[4] Feigenbaum, E. and Buchanan, B., "DENDRAL and META-DENDRAL: Roots of knowledge systems and expert system applications", *Artificial Intelligence*, Vol. 59, No. 1, 1993, pp. 233-240.

[5] Hart, A., Experience in the use of an inductive system in knowledge engineering, In Martin Merry, editor, *Expert Systems 86*, pp. 117-126, Cambridge University Press, 1986.

[6] Hunt, E. and Stone, P., *Experiments in Induction*, Academic Press, 1966.

[7] Lee, J., Zylka, M. J., Anderson, D. J., and Burdette, J. E. "A steroid-conjugated contrast agent for magnetic resonance imaging of cell signaling", *Journal of the Ame-*

rican Chemical Society, Vol. 127, No. 38, 2005, pp. 13164-13166.

[8] Michalski, R. and Chilausky, R., "Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis", *International Journal of Policy Analysis and Information Systems*, Vol. 4, No. 2, 1982.

[9] Michalski, R., A theory and methodology of inductive learning. In R. Michalski, J. Carbonell, and T. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pp. 83-129, Tioga Publishing, 1983.

[10] Michalski, R. and. Chilausky, R., "Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis", *International Journal of Policy Analysis and Information Systems*, Vol. 4, No. 2, 1980, pp. 125-161.

[11] Michalski, R., *A theory and methodology of inductive learning*, Springer Berlin Heidelberg, 1983.

[12] Mitchell, T., The Need for Biases in learning generalizations. Technical Report, Rutgers University, Dept. of Computer Science, 1980. Technical Report No. CBM-TR-117.

[13] Mitchell, T., "Generalization as search", *Artificial Intelligence*, Vol. 18, No. 2, 1982, pp. 203-226.

[14] Rendell, L., "A new basis for state-space learning systems and a successful implementation", *Artificial intelligence*, Vol. 20, No. 4, 1983, pp. 369-392.

[15] Samuel, A., "Some studies in machine learning using the game of checkers", *IBM Journal of research and development*, Vol. 44, No. 1.2, 2000, pp. 206-226.

[16] Szczepanik, W., Arciszewski, T., and Wnek, J., "Empirical performance comparison of two symbolic learning systems based on selective and constructive induction", Proceedings of the IJCAI-95 Workshop on Machine Learning in Engineering, 1995.

Author Profile

Sung K. Kim

Sung K. Kim is a professor of business administration at Chung-Ang University. He received his Ph.D. in Information Systems from New York University. He has been actively involved in advisory roles on Korea's national ICT planning and governance as well as major private firms' IT projects. He currently holds an appointment of Managing Director in the Korea CIO Forum. His current research interests include Enterprise Architecture, IT planning, and technology adoption.