PROJECT PROPOSAL

Portfolio optimization of liquid assets through aggregated expert advice from populations of competing investor agents.

Program:	Industrial Economics and Technology Management
Profile:	Managerial Economics and Operations Research
Student:	André Christoffer Andersen
Partner:	Stian Mikelsen (IME-IDI)
Date:	2011-04-16
Last updated:	2011-05-01

ABSTRACT

This project (and future thesis) proposal outlines a portfolio optimization framework that generates and aggregates portfolio advice from multiple weak experts, i.e., Investor Agents, in to a single coherent and well-optimized portfolio. The final portfolio aims to be applicable as real-world portfolio advice. The framework uses genetic algorithms to gradually improve populations of weak experts, and uses boosting in order to aggregate the surviving expert's advice. The Investor Agents are treated as black boxes and can be implemented using any number of methods, may it be, technical analysis, neural networks or simple rules of thumb. The project will mainly focus on how to represent temporal market information in a manner that the Investor Agent can process as well as actually implementing some reasonable Investor Agent. Future thesis work will focus on finalizing the Investor Agents and tying the advice together through boosting or similar methods.

Introduction

Knapsack problems are one of the most prevalent optimization problems you can find. The epitome of such problems is, of course, the portfolio optimization problem within liquid markets such as the stock market. It is a problem domain of extensive research and practical importance. The challenges that ensue from such research have made some researchers declared stock markets as near-perfectly efficient. Over time it is a zero sum game – you can't beat the stock market, they say.

I find this notion pessimistic and wish to contribute to the ongoing research by merging many of my own and my partner's fields of study in to a hierarchical framework for dynamic portfolio optimization. The framework will take temporal market information as input and then output a portfolio suggestion that should yield at least better return than the market itself.

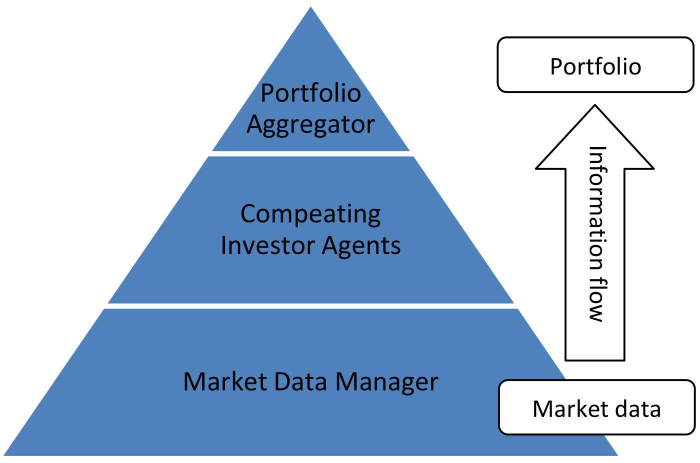
In essence the project we are proposing aims to lay the foundation for a general portfolio optimization framework that we wish to finalize as a joint master's degree thesis. For the rest of this proposal I will give a brief summary of the entire framework then expand on this project's scope within the framework.

Components of the Framework

The framework will be structured hierarchically with three major components:

- Market data management
- Investor agent competition
- Portfolio aggregation

The process information flow in broad terms: Market information is presented to the Market Data Manager which passes scrubbed data to a population of Investor Agents. These Investor Agents are in essence small computer programs that compete under evolutionary principles yielding better investors every generation. Finally, all the individual portfolios generated by the Investor Agents are passed to the Portfolio Aggregator. This component combines the portfolio into a single coherent optimized portfolio which is the final output to the end user.



Market Data Management

The raw marked data must initially be pre-processed before passing it to the Investor Agents. The very first task is using human intelligence in order to select reasonable features (data sources) that could be predictive of future market performance. The features need not be discriminately selected since the learning process will ignore uninformative data automatically. That is, too many features are better than too few. When there is a sufficient amount of features they need to be cleaned, transformed and normalized. Cleaning the data encompasses identifying incomplete, incorrect, inaccurate, irrelevant, noisy, etc. parts of the data and then replacing, modifying or deleting this dirty data. Transforming the data is another important step that aims to format the data in a way that the Investor Agents can best extract information. An example could be to calculate the P/E-ratio and presenting this as its own feature. Normalization only refers to scaling the data across all the features such that they have roughly the same mean and magnitude. Finally, a crucial part of data management is to represent time in a meaningful yet computationally tractable way. What time horizon and time resolution should we look at? Should new data be emphasized and let old data be more roughly represented? How do we handle feature drift (or changing data trends if you will)? Another, temporal problem with market data is the introduction and removal of new features which happens often in stock markets. A typical example would be when companies enter or exits the market for whatever reason.

COMPETING INVESTOR AGENTS

The main computing unit of the framework is the Investor Agent. These agents are small computer programs that, at their most rudimentary definition, take a subset of the available market features as input and outputs a specific portfolio suggestion. This definition is intentionally vague, partly because we haven't yet settled on preferable inner workings of the agents, but also because we would like to be able to populate the agent ecology with different types of agents without worrying about framework compatibility. Though, there are some evolutionary characteristics they must possess. They must be able to reproduce with variation and have inheritable traits that define their portfolio selection fitness. If for example we implemented the agents as neural networks we could code its properties (e.g, number of hidden layers, hidden layer width, connection degree, etc.) as genetic traits that would evolve over generations to an optimal configuration. As hinted at, for evolutionary algorithms to work there must exist some way to determine portfolio selection fitness, but not only that, there must also exist some agent death criteria. This can be implemented directly for each agent by applying actual market movements on their individual portfolios. This would be implemented by giving each new agent a finite single allowance which they virtually invest using their own portfolio suggestion. If the allowance runs out the agent is eliminated from the ecology. This, naturally, stops the agent from (further) reproduction thereby ending that particular evolutionary branch.

PORTFOLIO AGGREGATION

In of itself a population of Investor Agents should, just by genetic pressure and learning, do well on their own, however, constructing a single portfolio from them shouldn't be handled trivially. For instance just taking the mean investment of all agents would at best perform as well as the general John Doe agent, while we want the performance of the best agents. There exist, luckily, methods of performing almost as good as the best Investor Agent in hindsight. These methods are called boosting algorithms. They take weak learners (e.g., Investor Agents) and create a single strong learner that outperforms the average Investor Agent. The main criteria is that the individual Investor Agents perform just marginally better (or even worse would do) than selecting a portfolio at random.

As a final facet of the Portfolio Aggregator we would like to be able to impose some configurations to the results that it generates. For instance, having an upper and lower bound on the number of investment objects at any time, and likewise an upper bound on the number of transactions per unit of time, is important to make the framework practically usable as investment advice in the real world.

Computational complexity

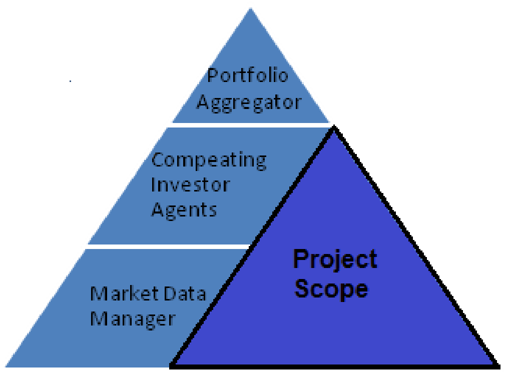
Because the Investor Agents are discrete computational units they can be implemented on separate memory and CPUs, or even separate computers altogether. This fact will greatly improve the computational outlook of the problem. The framework can thus be implemented on a networked server farm where several slave computers process the actions of discrete Investor Agent populations who passes the resulting portfolios to a master server where the Portfolio Aggregator resides.

The Project Scope

The scope of this project will be centered on the two lower levels of the framework hierarchy, i.e., the Market Data Manager and the Investor Agents. Some potential millstones are as follows:

- Market Data Manager
    - Extracting a rudimentary market data set
    - Prepared the market data for testing purposes.
    - Handling new and elimination of features.
- Investor Agents
    - Create a general interface for Investor Agents. This needs to be created with the intent of applying genetic pressure on it.
    - Some actual Investor Agents must be implemented. For instance,
        - Neural networks
        - Classical investment strategies
        - Bayesian Markov chain
    - The evolutionary algorithms must be settled.

Thus, the main goal is to create a healthy and sustainable population of Investor Agents that generate portfolios which can be trivially aggregated to produce at least marginal better performance than random portfolios. Future iterations of the framework will aim at outperforming the market itself.


Portfolio Aggregator
Compeating Investor Agents
Market Data Manager
Project Scope