Solving NP-Hard Problems: An Example of Universal Portfolios

Jenny Hung

October 26, 2017

T he portfolio selection is an interesting problem: it can be considered as a combinatorial optimization, a variation of the knapsack problem, or as a Mean-Variance optimization problem, or as a neural network problem. It can be solved in a variety of ways, using an adapted algorithm from the knapsack problem, or as weights for the neural network.

In this talk, we will explore the practical solution of the portfolio selection problem from two perspectives: first as an adapted knapsack problem where the choices of investment were made, then investment weights are computed using a neural network; the second approach leverages information theory to find a Bayesian estimator that gives weights to the investments directly.

Machine Learning Problem

The emphasis of machine learning is on automatic methods. In other words, the goal is to devise learning algorithms that do the learning automatically without human intervention or assistance. The machine learning paradigm can be viewed as programming by example. Often we have a specific task in mind, such as spam filtering. But rather than program the computer to solve the task directly, in machine learning, we seek methods by which the computer will come up with its own program based on examples that we provide. Machine learning is a core subarea of artificial intelligence. It is very unlikely that we will be able to build any kind of intelligent system capable of any of the facilities that we associate with intelligence, such as language or vision, without using learning to get there. These tasks are otherwise simply too difficult to solve. Further, we would not consider a system to be truly intelligent if it were incapable of learning since learning is at the core of intelligence.

More specifically, in the power set of all the hypotheses that exit, the goal of a machine learning problem is to find the best hypothesis that consistently explains the observed data. Then using this best hypothesis, then we can do all sorts of fancy things such as predict outcomes and prescribe actions in an intelligent way.

Bayesian Learning: Gold Standard

An astute observer should recognize the statement of **finding the best hypothesis that consistently explains the observed data** can be conveniently described mathematically as a decision rule that minimizes the posterior expected value of a loss function. An alternative way of formulating an estimator within Bayesian statistics is maximum a posteriori estimation.

The Universal Portfolio algorithm by Cover (1991) is indeed a Bayesian estimator, and is shown to generate a wealth process that is invariant under permutation of performance vector sequence. Therefore, the wealth produced by Cover?s universal portfolio is the same regardless of the distributions ?down-days? throughout the investment window. Moreover, it can also be shown that this algorithm produces wealth which exceeds that of value line index.

The biggest drawback of the universal portfolio algorithm is that in general, it takes a long time for the estimator to learn the weights.

Randomized Optimization Algorithms

We can also think of the portfolio selection as a combinatorial optimization exercise: given a set of investment choices, each with a return and volatility, determine the number of each investment to include in the portfolio, so that the total volatility is less than the tolerated level. When phrased this way, this problem has many other representations and has been studied in fields such as combinatorics, computer science, complexity theory, cryptography and, in general, applied mathematics.

The knapsack problem is interesting from the perspective of computer science for many reasons:

- The decision problem form of the knapsack problem (Can a value of at least V be achieved without exceeding the weight W?) is NP-complete, thus there is no known algorithm both correct and fast (polynomial-time) on all cases.
- While the decision problem is NP-complete, the optimization problem is NP-hard, its resolution is at least as difficult as the decision problem, and there is no known polynomial algorithm which can tell, given a solution, whether it is optimal (which would mean that there is no solution with a larger V, thus solving the NP-complete decision problem).

There is a link between the ”decision” and ”optimization” problems in that if there exists a polynomial algorithm that solves the ”decision” problem, then one can find the maximum value for the optimization problem in polynomial time by applying this algorithm iteratively while increasing the value of k . On the other hand, if an algorithm finds the optimal value of the optimization problem in polynomial time, then the decision problem can be solved in polynomial time by comparing the value of the solution output by this algorithm with the value of k . Thus, both versions of the problem are of similar difficulty.

One theme in research literature is to identify what the ”hard” instances of the knapsack problem look like,[9][10] or viewed another way, to identify what properties of instances in practice might make them more amenable than their worst-case NP-complete behaviour suggests.[11] The goal in finding these ”hard” instances is for their use in public key cryptography systems, such as the Merkle-Hellman knapsack cryptosystem