Pattern Recognition by Neural Network Ensemble

Joseph Cestra, Babu Johnson, Nikolaos Kartalis, Rasul Mehrab, Robb Zucker
Pace University

Abstract

This is an investigation of artificial neural network ensembles and its application to predicting future trends of the stock market. We discuss the importance of neural networks and the theoretical benefit of using an ensemble. The stock market is a complex and volatile system which, if adequately forecasted, could yield profitable decisions for traders. Conventional linear programming models lack the structure for such a problem; however, neural networks are particularly well-suited because of their parallel and adaptive nature. Ensemble techniques combined with the cross-validation training methodology are used to produce the ideal results. The overwhelming factors that influence the stock market as well as the varying network structures, parameters, and algorithms prove difficult to master, but the results are a step in the right direction.

1. Introduction

1.1. Artificial Neural Networks

Even with the vast improvements in computing power, there are many problems not suitable for a linear programming paradigm. The conventional algorithmic approach relies on a set of clearly defined instructions to solve a problem. Unfortunately, this restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. Problem solving is at the heart of computer science so it is vital that we develop and analyze new computing models suitable for real-world problems. One such model is the Artificial Neural Network (ANN), which has the remarkable ability to derive meaning from complicated or imprecise data and can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. The ANN derives its design and inspiration from the human brain and emulates biological neural networks in both structure and function. Structurally, an ANN consists of a pool of highly interconnected processing units called neurons working in parallel to solve a problem. The main function of an ANN is to learn from input and produce output based on past observations. Artificial Neural Networks are powerful because they are adaptive systems that have the unique ability to capture complex relationships between inputs and outputs as well as learn from a set of data without any underlying assumptions. We are focusing our studies on the Multilayer Perceptron, which is a feed-forward neural network trained under a supervised learning algorithm. In order to train these systems we feed a large set of inputs and desired outcomes into the network upon which the network can adapt. After many iterations of training, the ANN is able to detect subtle patterns in large data sets and make predictions based on what it has learned by past observations. ANN’s have been successfully applied to broad spectrum of applications including: financial forecasting, targeted marketing, medical diagnosis, and voice recognition. The ANN gives us a new way to approach complex data intensive problems.

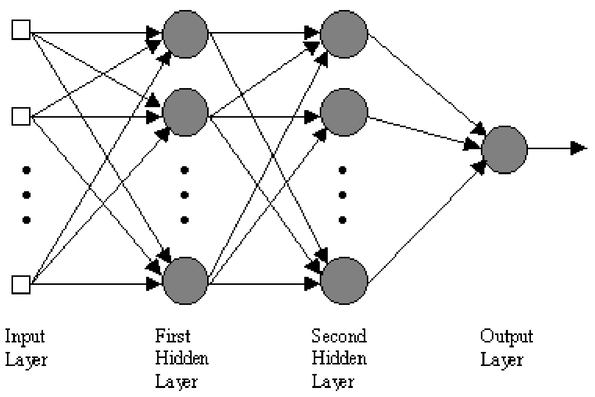

Figure 1. MLP Neural Network.

1.2. Ensembles

Ensemble techniques couple the output of a collection of multiple neural networks to form one collective decision. The critical issue in developing a neural network is generalization: how well will the network make predictions for cases that are not in the training set? In reality, the network is trained to minimize the error on the training set; however, this is not the same thing as minimizing the error surface of the underlying and unknown model. After all, the purpose is not to memorize a pattern for a given data set but; more importantly, to be able to generalize knowledge learned to unseen data. A network may be trained on a data set successfully, but when that network is tested on some new data the network may perform poorly. This means the network did not generalize effectively. Poor generalization can be a result of over-learning or over-fitting, which occurs when a network is too complex for a problem or given quantity of input. Over-learning most commonly occurs when the number of input variables (and hence the number of weights) is large with respect to the number of training cases [5]. The use of neural network ensembles is believed to improve generalization performance by combating over-learning. Statistically, this occurs because averaging predictions across models trained on different data subsets, can reduce model variance without increasing model bias [2]. Intuitively, dividing the input variables among an ensemble enables the number of training cases to remain constant while reducing the network complexity. Ultimately, theory suggests that the expected performance of an ensemble is greater than or equal to the average performance of the members [2] [5].

1.3. Objective

The vision for this project is to construct an ensemble of Multilayer Perceptron (MLP) Artificial Neural Networks and to evaluate its performance on the task of financial forecasting. Specifically, we hope to develop an ensemble of networks that are each trained individually on different sets of historical economic data, and then coupled to create a collective decision regarding buying, selling, or holding shares of stock. This kind of problem falls under classification, where a given input needs to be classified into a category. Since the decisions will be solely based on the direction and change of the market index, the network is ultimately predicting future trends of the stock market.

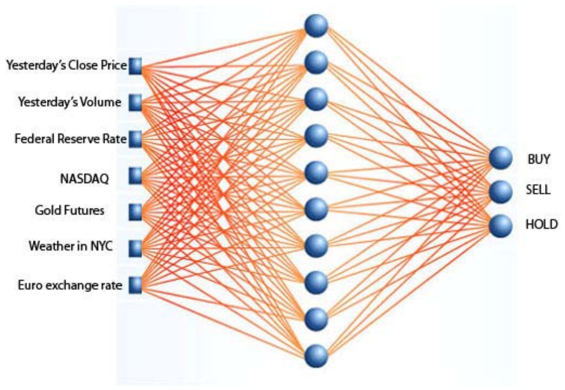

Figure 2. One member of ensemble.

Financial forecasting is of particular interest because it is a problem suitable for neural network ensembles and can be a lucrative endeavor. Other reasons include: unlimited access to historical data on the web and the existence of almost limitless inputs that affect the stock market. The stock market is an often erratic system that frequently fluctuates in response to many factors. Most programming paradigms are ill suited to make predictions in such a complex and volatile system. Our objectives include demonstrating that an ANN has the capacity to learn the behavioral patterns of the stock market and to show the benefits of using an ensemble of neural networks for collective decision-making. We believe that with a careful selection of inputs and design for our ensemble we can develop a system that makes favorable decisions in the stock market.

2. Design

We have chosen to use the Dow Jones Industrial Average for our investigation and, if we succeed, similar efforts will be applied to other stock indices in the future. Our initial ensemble consists of six networks each trained on a unique set of inputs. The output for each network will represent a decision to buy, sell, or hold stock. The data structure used for the output is an array of three binary values, where each value corresponds to a decision. A value of 1 is used to indicate a favorable decision and a value of 0 will indicate an unfavorable decision. The desired outcome for one trading day is based on the direction (positive or negative) the stock market went that day and the percentage of change. The threshold will be a 1% change in either direction meaning a market gain equal or greater than that amount will yield a buy decision, a loss equal or greater than that amount will yield a sell decision, and any percentage change less than that amount will yield a hold decision. A 1% change is equivalent to a 100 point gain or loss from a previous day closing of 10,000. Like many other factors this may be adjusted for future purposes. The corresponding values from each network output will be summed to create a single output and the greatest value of the output set will indicate the correct decision. For example, if two networks produce outputs [.99, .75, .23] and [.88, .99, .45] then the first item of the final output set would hold the greatest value and the corresponding decision would be made.

2.1. Training Data

The data used for training will consist of 21 input variables, which represent factors that are believed to affect the behavior of the stock market. The initial inputs chosen include the following:

DJIA Close	Nikkei Index
DJIA Volume	Hang Sang Index

Oil	NASDAQ
Gold	S&P 500
10 year Treasury Note	U.S Prime Rate
FTSE Index	Federal Funds Rate
DAX Index	Inflation
CAC 40 Index	Consumer Confidence
Dollar-GBP	Unemployment
Dollar-Yen	30 year Mortgage Rate
Dollar-Euro	

It is possible that some of the inputs will have greater impact on the stock market than others so it is useful to experiment with these values.

2.2. Data Division

Because we are using an ensemble of neural networks, the input data was divided amongst each network. Each network is fed with both a constant and unique input set. The constants data used for all networks consists of general economic data including bonds (10yr Treasury), commodities (oil and gold), and DJIA stock data (close and volume). The division of remaining training data was done using basic knowledge about the economy rather than by random. We coupled data that shared something in common to create unique sets of input variables that we believe would have a significant impact on the stock market. The division is as follows:

Network 1 consists of three exchange rates: USD-GBP, USD-Yen, and USD-Euro, which indicate the strength of the U.S dollar.

Network 2 consists of U.S market data and includes the closing values of two major indices; namely, NASDAQ and the S&P 500.

Network 3 consists of Asian Market data and includes the closing values of two major indices; namely, Hang Sang and Nikkei.

Network 4 consists of European Market data and includes the closing value of three major indices; namely, CAC 40, DAX, and FTSE.

Network 5 consists of factors which affect money supply and the purchasing power of that money. Inflation directly correlates to the purchasing power of the U.S Dollar while the Federal Funds and U.S Prime Rate both strongly influence the supply of money.

Network 6 consists of data which indicates the welfare of the general public. Unemployment has a direct impact on the consumer and its effects permeate through the economy. The Consumer Confidence Index indicates the attitude of the public toward the economy. The 30 year Mortgage Rate indicates the ups and downs of the housing market, whose conditions are critical to millions of homeowners and families.

With limited knowledge in economics and the stock market we hope we have coupled data in a way that has a meaningful impact on our network output and at the very least no negative effect.

2.3. Training Methodology

The purpose of training is to allow networks to acquire knowledge or learn from a sample of data but the success of training is dependent on how well that knowledge can be applied to data outside the training sample; otherwise, known as generalizing. In order to develop networks with best possible performance we need a way to accurately measure generalization error. Before we do that we need an appropriate error function to calculate or characterize the overall error of a network. We have chosen to use the root mean squared function to approximate the total error of a network for a given data set. Since our goal is to find the network having the best performance on new data, the simplest approach to the comparison of different networks is to evaluate the error function using data which is independent of that used for training. The most common approach requires subdividing the sample data into training, validating, and testing sets. Various networks are trained with respect to a training data set. The performance of the networks is then compared by evaluating the error function using an independent validation set, and the network having the smallest error with respect to the validation set is selected. The performance of the selected network should be confirmed by measuring its performance on a third independent set of data called a test set. The main criticism of this methodology is that the size of the training set is necessarily reduced. For this reason, we have opted to use cross-validation. We will divide the data into n subsets and train the net n times, each time leaving out one of the subsets from training, but using only the omitted subset to compute the error. This allows us to use all the data for training; however, we will still need an independent set for final testing.

3. Implementation

3.1. Programming

We adapted an existing neural network written in Python for our investigation. Python is rich in text processing tools, it is easily extensible, and supports object-oriented, procedural, and functional styles of programming. The program code employs a back-propagation learning algorithm and defines a network with one hidden layer.

3.2. Data Collection

The data was collected through the Web from sources such as Yahoo Finance and stored in the CSV (Comma Separated Value) format. CSV is the most common import and export format for spreadsheets and databases. Furthermore, Python has a built-in CSV module for reading and writing. The data was collected for one year and organized appropriately for the individual networks. To facilitate the training process, data for each network was organized in separate CSV files in order to limit the need for parsing the data in python.

3.3. Training, Validation, and Testing

Adjustments were made to the existing code to automate training from the collected data and each neural network was fed a unique subset of the training data. The initial parameters for each neural network included: 5 hidden neurons, 3 output neurons, and a .5 learning rate. We experimented with these and other parameters and compared network performance using the cross-validation method. After choosing the parameters that yielded the best generalization performance for each member of the network we tested the ensemble using an independent set.

4. Evaluation

4.1. Cross-Validation

We performed the cross-validation method on each individual network in order to select the proper amount of hidden neurons to be used. We calculated both statistical generalization error (using RMS) and the hit rate of the network. The results are as follows:

Hidden Neurons	Statistical Error	Hit Rate
5	.75	30%
7	.80	27%
9	.81	28%
10	.81	45%
12	.81	44%
14	1.1	28%

Table 1. Network 1 results.

Hidden Neurons	Statistical Error	Hit Rate
5	.75	27%
7	.80	27%
9	.81	28%
10	.81	45%
12	.81	44%
14	1.1	28%

Table 2. Network 2 results.

Hidden Neurons	Statistical Error	Hit Rate
5	0.75	27%
7	0.8	33%
9	0.81	27%
10	0.81	31%
12	0.82	45%
14	.85	27%

Table 3. Network 3, 4, 5, 6 results

The first two networks had the highest hit rate using 10 neurons, while the rest performed best with 12 neurons. The generalization error remained fairly stable using up to 12 neurons. When 14 or more neurons were used, the performance dropped off considerably for each network, which means the network was probably over-learning. We ultimately chose to use 12 neurons for each of the networks in the ensemble.

4.2. Ensemble Results

As described in the project design, we trained each network individually. Using a random sample of the data we tested each network and the individual output sets were combined. The ensemble output set was then compared to the desired results so that we could calculate both statistical generalization error as well as the hit rate.

5. Conclusion

The results were not as accurate as we would have hoped; however, we believe it is a step in the right direction. The complexity of the problem and of network design has served as a reminder that financial forecasting is no easy task. As far as the stock market goes, there are numerous factors that may or may not affect future trends. Selecting the ideal input variables can be a daunting task and coupling that data for the use of ensemble only adds to the complexity. It would be helpful to combine ANN research with studies that show the affects of economic data and other factors on the stock market. As far as ANN's go, there are numerous factors to contemplate: number of hidden layers, number of neurons, size of training sample, error function, training methodology, learning algorithm, network structure, data representation, ect. The wrong choices lead to poor performance. A small network may not learn what you want it to learn, while a large network will learn slowly, may get stuck on local maxima, and may exhibit over-fitting. Overall, developing ANN's is as much an art as a science. One must carefully select data samples and network design parameters and undergo thorough experimentation, in order to get the best results. The successful deployment of ANN technology requires time and experience. ANN experts are artists; they are not mere handbook users [4].

6. References

[1] Mertz, David, and Andrew Blais. "An introduction to neural networks: Pattern learning with the back-propagation algorithm". 1 July 2001. Available: https://www.ibm.com/developerworks/library/l-neural/.

[2] (Electronic Version): StatSoft, Inc. (2010). Electronic Statistics Textbook. Tulsa, OK: StatSoft. Available: http://www.statsoft.com/textbook/stneunet.html.

[3] Galkin, Ivan. "Crash Introduction to Artificial Neural Networks". Available: http://ulcar.uml.edu/~iag/CS/Intro-to-ANN.html.

[4] Fahey, Colin. "Neural network with learning by backward error propagation". Available: http://www.colinfahey.com/neural_network_with_back_propagation_learning/neural_network_with_back_propagation_learning_en.html.

[5] Warren S. Sarle. AI FAQ/neural Nets. Available: http://www.faqs.org/faqs/ai-faq/neural-nets/.

[6] http://finance.yahoo.com/.

[7] http://www.oanda.com/currency/historical-rates.