How could Machine learning algorithms such as Genetic Algorithms and Bayesian Networks lead to a crash in the Stock Market?

Computer Science

Jeffrey Worley

Word Count: 3721

May 2017 Examination Session

Table of Contents

Abstract

How could Machine learning algorithms such as Genetic Algorithms and Bayesian Networks lead to a Crash in the Stock Market? Recent decades have seen an economic revolution larger than has ever been observed in history. The addition of computers to the Stock market, starting with the NASDAQ in the 1970's (NYSE, 2016), has allowed for the volume of trades made in a day to explode. Furthermore, computers have allowed for many new strategies for investment in the stock market such as high-frequency trading. However, these may be the revolutions of the past as many have begun to look towards Machine Learning (more commonly known as artificial intelligence) as the next step for profiting from the stock market. Many hedge funds have already begun to implement Machine learning algorithms such as Genetic Algorithms and Bayesian Networks (Redmond, 2016). While many see this as an opportunity to profit, many worry that such reckless use of advanced technology could lead to economic disaster (Tegmark, 2016; Sainato, 2015; Cellan-Jones, 2014). Those showing concern reference the Flash crash of 2010 as proof that measures have to be taken to protect against this new agent in the stock market. My research, thus, aims to investigate the causes of the Flash Crash of 2010 and provide an analysis of how different Machine Learning algorithms work and how they could lead to events like the Flash Crash and/or how they could be resistant to the same event. And through my research into Genetic Algorithms and Bayesian networks, I have found that while they can perform in normal market situations, any sudden shift can lead to economic downfall due to the inflexible and non resilient nature of Machine Learning to new situations.

Word Count: 284

Introduction

In recent decades, the addition of computers to the stock market has revolutionized the economy by increasing the volume of trades per day on the New York Stock Exchange (the largest stock exchange in the world) from an average of fewer than 100,000 trades a day in the 1990's to an average of several million per day today (NYSE, 2016). This phenomenon is a byproduct of the massive amount of basic calculations that a computer can perform in a short period of time. Recent advancements in computing, specifically Machine Learning (more commonly known as Artificial Intelligence), may be the next revolution to come to the stock market. Many researchers theorize that Machine Learning algorithms could perform much better than a human can at predicting the stock market. Individuals such as AI researcher Junsuke Senoguchi have developed AI that can do just that. His AI boasts the ability to predict market movements 68% of the time (Redmond, 2016). This is quite the feat given that market experts can (on average) only predict market movements 48% of the time ("Guru Grades", 2016). While many see AI as the next great opportunity to cash in, others, such as Stephen Hawking, Elon Musk, and Bill gates (Tegmark, 2016; Sainato, 2015; Cellan-Jones, 2014), worry that Artificial intelligence could spell disaster for the financial sector and even human existence .

Most of these AI based trades come from a few of the most promising Machine Learning methodologies that have emerged over the course of the last few decades. Of the promising algorithms, the ones that have had the highest popularity in stock market applications are Genetic Algorithms and Bayesian Networks (Metz, 2016). Genetic Algorithms are founded on biological principles observed in nature. This algorithm aims to mimic the process of natural selection. These algorithms determine the best iterations of the program in a given generation and then choose, by various different methods, how to combine these iterations in ways that hope to produce stronger future generations (Koza, 1992). Bayesian Networks are instead based on probability theory, probabilistic logic, and Bayesian logic. Bayesian Networks use these branches of mathematics to create an AI that makes decisions by calculating probabilities for events and variables (Jensen, 2001).

Currently not a lot of research is publicly available for Machine Learning used in the stock market due to the proprietary nature of the research. The hedge funds that are doing well with their algorithms wouldn't dare share their data sets and methodologies for fear of lost profits and giving over advantages to other hedge funds. However, The information provided in several sources on the functionality of the algorithms themselves can shed light on how the usage of the algorithms works with the stock market.

This paper looks to answer the question: **How could Machine learning algorithms such as Genetic Algorithms and Bayesian Networks lead to a Crash in the Stock Market?** This paper will delve into the methodologies by which Genetic Algorithms and Bayesian networks work. From there I will discuss the different ways in which the different algorithms would be resistant and/or nonresistant to situations such as the Flash Crash of 2010. This will conclude by answering How could Machine Learning Algorithms such as Genetic Algorithms and Bayesian Networks lead to a crash in the stock market like the Flash Crash of 2010.

Genetic Algorithms

Genetic Algorithms are an attempt to replicate the biological process of evolution. Evolution is the biological process where individuals of a species with more desirable traits are better able to survive, compete, and reproduce. This, in turn, causes those traits to be passed on to the next generation. Genetic Algorithms employ evolution in the development of a program with the goal of creating the best solution to a given problem through generations. Each generation has several iterations (individuals) of the program that are offspring of previous generations. Each individual of a generation has its own set of unique characteristics represented by binary "chromosomes". Chromosomes being the biological way of storing traits of an organism. These chromosomes then undergo processes to attempt to imitate biological evolution. Notably, the biological processes of chromosomal crossover and mutation are mimicked (Galán, Mengshoel, & Pinter, 2013).

Chromosomal crossover is the method by which many organisms reproduce. Genetic Algorithms edit this process slightly by only considering the strongest individuals of the current generation of a program and then "breeding" those individuals together to try to produce even stronger future generations. The breeding in chromosomal crossover results in an iteration that has a random combination of the parents' chromosomes (Srinivas, 1996). An example would be if a program's chromosomes consisted of 4-bits and the program's goal was to create the binary representation of 15 (1111). Consider a situation in which the strongest iterations of the program in this given generation have the binary chromosome representations of "1101" and "1110." A random chromosomal crossover could result in an offspring with the chromosome "1111."

Mutation is a biological process where sometimes chromosomes are not perfectly copied from the parent to the offspring, increasing variety. This is the other approach to creating new generations of programs in Genetic algorithms. Each bit of a binary chromosome is given a probability of mutation. When offspring are produced, the parent chromosomes undergo this random change in bit values. Another important aspect of computational mutation is that there is only one parent individual as compared to chromosomal crossover (Srinivas, 1994). In regards to the same situation discussed previously, consider that there is a parent whose chromosome holds the value of "1001." A set of random mutations could occur creating an offspring with the chromosomal bits of "1111."

While there has been debate as to which of these two methods is superior, studies have shown that neither of these two methods is better at creating stronger iterations than the other (Senaratna, 2005).

Both of these methods rely on finding the strongest iteration of a generation. While there is no definite biological method to determine an individual's strength, Genetic Algorithms do have a method to determine an Individual's strength. An individual's strength in a Genetic algorithm is measured by what is referred to as a "fitness score". The fitness score is a measure of how good an individual is at performing a desired task. The fitness score for a specific program is defined by the developer of the program. The fitness score, then, creates a numerical value to compare individuals against each other for the purpose of breeding the next generation (Mitchell, 1996). For clarification, an example can be found in the AI MarI/O. Developed by Computer scientist Seth Bling, MarI/O uses Genetic Algorithms to create a program that teaches itself to complete a level of the popular video game, Super Mario Bros. Unlike most other AI's, Seth Bling has released the source code of this program for public view, allowing analysis. From reading the source code, the fitness score increases as the program moves the character forward in the level (along with a few other factors that require knowledge of the game). As a new generation of this program is being made, the iterations that made it farthest into the level (the individuals with the best fitness score) are bred together to produce the new generation of the program (Bling, 2015).

The general approach to a Genetic algorithm starts out with a randomized generation. The Genetic Algorithm has different functions that It can use, but the random initial iterations randomly choose what functions are called (Bling, 2015; Koza, 1992). Genetic algorithms often start out with doing nothing . This is due to the program not being preprogrammed to do what it should be doing. This is considered good practice because it allows for the program to explore any possible solution instead of having to start with what the developer thinks the solution is. This often occurs for many generations until an evolution occurs and causes there to be a jump in fitness score. This beginning fitness score jump creates a starting place for a genetic algorithm to start breeding from. This process continues producing generations until the algorithm finds what it determines to be the best iteration of the program (Koza, 1992).

Genetic Algorithms have gained popularity through their nonlinear approach to a solution. Problems with nonlinear solutions do not have a single maxima like linear problems do. This means that nonlinear problems have many good solutions due to the presence of many relative maxima. Genetic Algorithms are inherently nonlinear because evolution does not strictly increase towards a finite solution; each new generation is not necessarily stronger than the last. Instead, evolution works towards finding the different maxima that exist for a problem (Yokota, Gen, Li, & Kim, 1996). An example of this would be when NASA used Genetic Algorithms to develop the best antennae shape for a spacecraft in 2006. The resulting shape looked like a random bending of a wire. This demonstrates the nonlinear problem of antennae design. (Hornby, Globus, Linden, & Lohn, 2006)

However, Genetic algorithms often run into “local peaks.” This can be thought of as instructing a person to reach the top of a mountain blindfolded. The person would walk until he/she started walking down. The person would assume that this peak they’ve reached is the peak (Jacobson, 2012). However, how does one actually know if this is the global peak? This problem actually doesn’t have an answer. To be exact, the only answer available is that it is impossible to know if one is in a local peak as found by the Now Free Lunch Theorem (Ho & Pepyne, 2002). However, the accepted method of working around this issue is to promote generations that attempt to move away from what is currently determined as the best solution. This is done by giving a penalty to the fitness score of iterations of the program that are too similar to an already determined peak (Beasley, Bull, & Martin, 1993).

Bayesian Networks

Bayesian Networks aim to provide solutions using probabilities. Named after Thomas Bayes, Bayesian Networks rely on probability theory, bayes theorem/probabilistic logic, and Inductive learning. These principles culminate in Bayesian Networks representation of variables and their relationship with each other through Joint Probability models. Bayesian networks are very good at predicting future events given information about past events and their outcomes by dealing with the relationships between variables as calculated through these methodologies (Koski & Noble, 2009; Pearl, 2011).

Probability theory is a field of mathematics focused on the chances that random events will occur, the probabilities. Probability theory provides a way of describing these random events concerning how likely they are to occur. The classic example being the probability of a coin landing on either heads or tails. Because there are two outcomes both with equal chance of occurring (in a world where the weight of a coin is perfectly balanced), 50% of the time the coin will land on heads, and 50% of the time it will land on tails. However, obviously there are circumstances where one could flip a coin 5 times and get 5 heads or 5 tails in a row. There is a law associated with probability theory that provides reliability of probabilistic calculations. This being the law of large numbers. This basically states that as sample size increases, the occurrences of random events will approach their expected probabilities (Kolmogorov, 1956) .

Bayesian theorem and probabilistic logic are related in the sense that they both use probabilities to make assumptions about the unknown. Bayes theorem is:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

In layman’s terms, this theorem states that the probability of event A occurring given event B has occurred is equal to the probability of event B occurring given event A has occurred multiplied by the probability of A. All of this is then divided by the total probability of event B occurring (Vapnik, 2000). To clarify, I will use the example of a test for cancer with random numbers for the probabilities. The probability of someone having cancer is 1%. The probability that a person who has cancer is diagnosed with cancer is 80% with a 20% chance for testing negative. The probability that a person doesn’t have cancer is 99%. The probability that a person who does not have cancer will test positive is 10% with a 90% chance of testing negative. With this we can calculate The probability one has cancer given a positive result represented as P(C|P).

$$P(C|P) = \frac{.8 * .01}{.8 * .01 + .1 * .99} = .0747 = 7.5\%$$

So, through bayes theorem, we can find the probability that a patient actually has cancer given a positive test result. With these random numbers, we calculated a 7.5% chance of having cancer given a positive test result. This is where probabilistic logic comes into play. Probabilistic logic uses probabilities to make decisions. In this case, if a program were deciding whether a person had cancer given only this information, the program would logically conclude that the person does not have cancer, because the chances of an individual having cancer even with a positive result is very low, in this random scenario (Vapnik, 2000 & Nilsson, 1986).

Inductive learning is how Bayesian Networks actually apply these methods to be considered a Machine Learning Algorithm. Inductive learning seeks to create a general rule for a given problem. It approaches this by using data sets of past events and their outcomes to generate the probabilistic relations between different variables of a program (Russell, Norvig, Canny., Malik, and Edwards, 2003). This results in a program that is not certain as to how to solve a problem, but the whole reasoning for applying Bayes theorem and probabilistic logic is to use uncertainty to the advantage of the program. Instead of finding precise answers like Genetic Algorithms, it creates predictions based on these calculations, and these predictive programs work very well for data they've analyzed. However, this means that Bayesian networks are subjective in that their probabilities work great with previously analyzed data sets, but struggle with the unfamiliar if the data sets are not large enough or diverse enough to create a robust AI (Niedermayer, 2008; Russell, Norvig, Canny., Malik, and Edwards, 2003).

Algorithms in the Stock Market

Now that the inner workings of each of the popular Machine Learning methodologies that are applied to the stock market have been investigated, I will discuss what aspects about them make them particularly strong in this application and how their weaknesses could be disastrous by investigating how they would have reacted to the Flash Crash of 2010.

The Flash Crash of 2010 is currently the only market crash that was only caused by automated trading. Specifically, the blame is placed on an investor who spoofed the E-mini S&P 500 futures market by using a automated trader algorithm he developed to always attempt to sell a stock at a price higher than anyone would buy for in extreme volumes (Levine, 2016). This spoofing of the market led to 1 trillion dollars suddenly disappearing in the less-regulated hours of the aftermarket (active trading hours have safeguards to protect against such a market dip caused by automated trading). This extreme fall in market share was caused by other computer and AI based investors reacting to the sudden surge in the volume of sale attempts in the market. This, in turn, caused the overall price to plummet due to these automatic investors concluding that others are trying to sell, so they should sell too. Correspondingly, The algorithms began to sell stocks in a very high volume, artificially driving the price well below its actual market value. In mere minutes, the market had seen a crash much larger than the 2008 market crash. While many investors began to try to figure out what they would do with their lives after losing their job, the market had nearly fully recovered in a matter of minutes with no explanation. (Frey, 2016). Given these events, it can further be investigated how Genetic algorithms and Bayesian networks could cause or be resistant to a crash such as this.

Genetic Algorithms' ability to evolve with constant reiteration is crucial to their success in the stock market. While the market has patterns, these patterns are not necessarily reliable, or the 48% average accuracy rate amongst experts would be drastically higher ("Guru Grades"). This unreliability can be traced to the fact that market trends are always changing due to an uncountable amount of factors that go into an economy. This means that an automated investor with the ability to identify what currently is the prevalent factor in the market will be able to predict trends better than a human analyst. This, additionally, means that the stock market is a perfect example of a nonlinear problem. The stock market has aspects that want to be maximized (such as profits) and aspects that want to be minimized (such as risk). This set of inequalities make the stock market a near exact definition of a nonlinear problem. A Genetic Algorithm would, with large enough data sets and enough processing power, be able to create a very robust and dynamic solution to stock market applications (Kim & Han, 2000; Dey, Kapoor, & Khurana, 2011).

However, there are some aspects of Genetic Algorithms that weaken it in terms of stock market applications. The stock market is a prime example of how Genetic algorithms could run into problems of local maximums of fitness scores. A Genetic Algorithm may place a large amount of focus on one variable of the stock market and it may find success and continue breeding generations that focus on this single variable. This could lead to the program getting stuck in a local peak. While methods do exist to work against this issue, the Stock market has so many variables that the Genetic Algorithm could foreseeably never find a global peak solution. (Allen & Karjalainen, 1998).

This is where Genetic Algorithms could have had a hand in the Flash Crash of 2010 if any were being used. Comparatively, Genetic algorithms are resistant to changes in market trends due to their constant evolution. However, extreme, sudden shifts in the market could spell temporary disaster Genetic Algorithms would have been slightly resistant to an event like this, especially if it had been fed data sets from the 2008 market crash. However, the Stock market crash of 2008 was not similar to the Flash Crash. The 2008 stock crash happened over the course of weeks whereas the flash crash happened in a span of about 30 minutes. The Genetic Algorithm based investors would have had no time to evolve, and would only have their current iteration to work with.Thus, while Genetic Algorithms are relatively strong in terms of normal market terms, any extremely sudden change could practically destroy the market with enough Genetic Algorithms reacting in the complete wrong way (Kim & Shin, 2007).

In contrast, Bayesian networks strengths in stock market applications lie in its processes of evaluating probabilistic logic to make decisions. The different variables are weighed as determined by the inductive learning that a Bayesian Network would have been subjected to. These probabilities would be very strong indicators of how the stock market is going to change, because that is basically how trading works from day-to-day. The different variables are weighted by analysts and the ones found to be important hold more weight in the decision on what to buy and sell. Thus, the Bayesian Networks would really succeed at finding the relationships between variables and their influence on the market (Jangmin, Lee, Park, & Zhang, 2004).

However, Bayesian Networks would also fall behind in shock events and would have a hard time adapting to market trends. When Bayesian Networks are designed, they work really well for events and data sets that they have analyzed and adjusted the algorithm for. If the Bayesian Network is not actively attempting to process current market trends and take those into account of their probability weights of variables, the program would quickly lose its ability to perform well in the stock market. Even if this were taken into account and the Bayesian Network was actively being improved and reiterated, The program would lose nearly all of its applicability in the event of a market shock. Even as drawn out of an event as the 2008 stock crash would cause a Bayesian Network to quickly fall apart (Margaritis, 2003).

Specifically, in the Flash Crash, a Bayesian Network is a prime culprit for automated investors that made the wrong decisions due to the spoofing. The Bayesian Network wouldn't have had any kind of resilience against an event like this, as this was the first time such a sudden market crash had occurred in the history of human markets. There would be no variables or probability values that would have had any kind of application to an event like this. As talked about in the Introduction, the most likely action of a Bayesian Network in this event would be to sell everything possible to minimize loss, as the configuration of the AI at the time of the event could have only determined that losses were inevitable, and selling was the only option (Shenoy & Shenoy, 2000).

Conclusion

In reference to my research question (**How could Machine learning algorithms such as Genetic Algorithms and Bayesian Networks lead to a crash in the Stock Market?)**, My investigation has shown that the current popular Machine Learning algorithms, Genetic Algorithms and Bayesian Networks, can perform in the stock market to an extent. And their preprocessed preparation for application to the stock market could even perform extremely well given the right situation. However, The methodologies are not without fault. Due to their inability to quickly react to market shift, these algorithms can lead to events like the Flash Crash of 2010. The Flash Crash of 2010 is a perfect example of why AI based automated simply can not be unregulated. So while regulations exist to prevent events during market hours, there needs to be the addition of regulations in the after market trading hours of the stock market. In addition, research into these Machine Learning algorithms needs to be expanded to create stronger methodologies to create algorithms that could one day be robust enough to perform well in any situation of the stock market.

Bibliography

Allen, Franklin, and Karjalainen, Risto. "Using Genetic Algorithms to Find Technical Trading Rules." *Journal of Financial Economics* 51 (1998): n. pag. Web.

Beasley, David, David R. Bull, and Ralph R. Martin. "An Overview of Genetic Algorithms : Part 1, Fundamentals." *University Computing* 15(2) (1993): 58-69. Print.

Bling, Seth (2015) MarI/O (Version 1.0) [Source code]. http://pastebin.com/ZZmSNaHX

Cellan-Jones, Rory. "Stephen Hawking Warns Artificial Intelligence Could End Mankind." BBC News, 2 Dec. 2014. Web. 02 Oct. 2016.

Dey, S., V. Kapoor, and A.P. Khurana. "Genetic Algorithm: An Application to Technical Trading

System Design." *International Journal of Computer Applications* 36.5 (2011): n. pag. Web.

Frey, Thomas. "Artificial Intelligence Will Be Crashing the Stock Market in 3, 2, 1..." *DaVinci Institute – Futurist Speaker*. Thomas Frey, 2014. Web. 19 Sept. 2016.

Galán, Severino F., Ole J. Mengshoel, and Rafael Pinter. "A Novel Mating Approach for Genetic Algorithms." *Evolutionary Computation* 21.2 (2013): 197-229. Web.

"Guru Grades" CXO Advisory Group LLC., n.d. Web. 02 Oct. 2016.

Ho, Y. and Pepyne, D. "Simple Explanation of the No-Free-Lunch Theorem and Its Implications." *Journal of Optimization Theory and Applications* 115.3 (2002): 549-70. Web.

Hornby, Gregory, Al Globus, Derek Linden, and Jason Lohn." Automated Antenna Design with Evolutionary Algorithms." *Space 2006* (2006): n. pag. Web.

Jangmin, O., Lee, J. W., Park, S. B., & Zhang, B. T. "Stock trading by modelling price trend with dynamic Bayesian networks." *International Conference on Intelligent Data Engineering and Automated Learning*. Springer Berlin Heidelberg, 2004.

Jensen, Finn V. "Bayesian Networks and Decision Graphs". New York: Springer, 2001. Print.

Jacobson, Lee. "Creating a Genetic Algorithm for Beginners." *The Project Site*. N.p., 12 Feb. 2012. Web. 15 Oct. 2016.

Kim, Hyun-jung, and Shin, Kyung-shik. "A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets." *Applied Soft Computing* 7.2 (2007): 569-576.

Kim, Kyoung-jae, and Ingoo Han. "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index." *Expert systems with Applications* 19.2 (2000): 125-132.

Koski, Timo, and John M. Noble. "Bayesian Networks: An Introduction." Chichester, West Sussex, UK: Wiley, 2009. Print.

Kolmogorov, A. N. "Foundations of the Theory of Probability." New York: Chelsea Pub., 1956. Print.

Koza, John R. "Genetic Programming: On the Programming of Computers by Means of Natural Selection." Cambridge, MA: MIT, 1992. Print.

Levine, Matt. "Guy Trading at Home Caused the Flash Crash." Bloomberg, n.d. Web. 04 Oct. 2016.

Margaritis, Dimitris. "Learning Bayesian Network Model Structure from Data." *School of Computer Science - Carnegie Mellon University* (2003): n. pag. Web.

Metz, Cade. "The Rise of the Artificially Intelligent Hedge Fund." Wired, 25 Jan. 2016. Web. 04 Oct. 2016.

Mitchell, Melanie. "An Introduction to Genetic Algorithms." Cambridge, MA: MIT, 1996. Print. NYSE. "TRANSACTIONS, STATISTICS AND DATA LIBRARY." *NYSE: Transactions, Statistics and Data Library*. NYSE, n.d. Web. 01 Oct. 2016.

Niedermayer, Daryle. "An introduction to Bayesian networks and their contemporary applications." *Innovations in Bayesian Networks*. Springer Berlin Heidelberg, 2008. 117-130.

Nilsson, Nils J. "Probabilistic Logic." *Computer Science Department, Stanford University.* (1986): n. pag. Web.

Pearl, Judea. (2011). "Bayesian networks". *Department of Statistics, UCLA*. UCLA: Department of Statistics, UCLA.

Russell, S. J., Norvig, P., Canny, J. F., Malik, J. M., and Edwards, D. D. "Artificial intelligence: a

modern approach." Vol. 2. Upper Saddle River: Prentice hall, 2003.

Sainato, Michael. "Stephen Hawking, Elon Musk, and Bill Gates Warn About Artificial Intelligence." Observer, 19 Aug. 2015. Web. 02 Oct. 2016.

Senaratna, Nuwan I. *Genetic Algorithms: The Crossover-Mutation Debate*. Rep. Computer Science, University of Colombo. N.p.: n.p., 15 Nov. 2005. Print.

L.m. Patnaik., and Srinivas, M. "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms." *IEEE Transactions on Systems, Man, and Cybernetics* 24.4 (1994): 656-67. Web.

Redmond, Tod. "This Japanese Robot Calls the Market Better Than a Human." *Bloomberg.com*. Bloomberg Technology, 17 Feb. 2016. Web. 19 Sept. 2016.

Shenoy, Catherine, and Shenoy, Prakash P. . "Bayesian network models of portfolio risk and return." (2000).

Tegmark, Max. "AI Open Letter." *AI Open Letter*. Future of Life Institute, n.d. Web. 02 Oct. 2016.

Vapnik, Vladimir Naumovich. *The Nature of Statistical Learning Theory*. New York: Springer, 2000. Print.

Yokota, Takao., Gen, Mitsuo., Li, Yinxiu., and Kim, Chang Eun. "A genetic algorithm for interval nonlinear integer programming problem." *Computers & industrial engineering* 31.3 (1996): 913-917.