HIGH SCHOOL EDITION
Journal of Student Research

Volume 9 Issue 2 (2020)
Research Article

Approaching Stock Market Trading with Echo State Networks

Dev Patel[1], Krish Patel[1] and Charles Dela Cuesta[1]

[1]Thomas Jefferson High School for Science and Technology/Alexandria, VA, USA

ABSTRACT

The US stock market is an integral part of modern society. Nearly 55% of Americans own corporate shares in the US stock market (What Percentage of Americans Own Stock?, 2019), and as of June 30th, 2020, the total value of the US stock market was over 35 trillion USD (Total Market Value of U.S. Stock Market, 2020). The stock market is also extremely volatile, and many people have gone bankrupt from poor investments. To minimize the risk and capitalize off the massive amounts of data on corporations and share prices present in the world, algorithmic trading began to rise. Trading algorithms have the potential for huge returns, and while many algorithms employ strategies like Long-Short Equity, very few attempt to use machine learning due to the unpredictable nature of the stock market. Many time series prediction models like autoregressive integrated moving average (ARIMA), and even neural networks like long short term memory (LSTMs) often fail when predicting stock market data, because unlike other time series data, the stock market is almost never univariate, or follows seasonal trends. However, where other models come short, echo state networks (ESNs) excel, due to their reservoir like computing model, which allows them to perform better on messy, non traditional time series data. Using a combination of ESNs to predict prices, and clustering we created an algorithm model that can predict trends with over 95% confidence, but had mixed results accurately predicting returns.

Echo State Networks

The Echo State Network is a subset of a Recurrent Neural Network (RNN). A RNN is defined as "any network whose neurons send feedback signals to each other" (Grossberg, 2013). In Computing, Artificial Recurrent Neural Networks (aRNN) are often used to create high powered, complex, interconnected models. We can oversimplify to describe the basics of how aRNNs function. Let's create an example aRNN. Our example aRNN has 3 layers - an Input, Processing, and Output Layer. Our aRNN has three inputs, meaning that our input layer has 3 neurons. Our Processing layer has 4 neurons, with each neuron performing a specific task. Our output layer has 1 neuron. Different Input neurons connect to various combinations of processing neurons, and each processing neuron connects to the output neuron, which takes the results from the processing layer, and translates it into the output we want. The different paths each neuron takes creates a kind of net, hence why it's called a neural network.

While our example model only has about 8 neurons in total, most neural networks are much more complex. They can be used for many tasks, including Time Series Prediction. The main issue with using regular Neural Networks for Time Series Prediction and other complex models is something called a disappearing gradient. RNNs optimize themselves using a technique called Backpropagation (Stewart, 2020). Backpropagation works by calculating error, checking to see if the error is minimized, and updating parameters and weights on the neurons if necessary (What is Backpropagation?, 2017). The disappearing gradient problem arises when the weights on neurons in a network get increasingly close to 0 due to overfitting and optimization of backpropagation. Neural Networks cannot properly learn if neurons have weights of 0 (the neuron is essentially skipped in the learning process).

Scientists have worked around this problem by creating special types of Neural Networks, like the Long Short Term Memory Network (LSTM). LSTMs have a "leaky unit" and store previous weights in "long term memory" which allows them to not be affected by disappearing gradients. While certain Neural Networks like the LSTM can be used to forecast Time Series Data, they are normally used for stationary, and seasonal data. When forecasting with Stock Market Data, LSTMs have had varying degrees of success. Certain studies reported a Root Mean Square Error (RMSE) between LSTM Predictions and Real values as low as .1241 (Kim and Kim, 2019).

While both are subsets of the RNN, LSTMs and ESNs are very different. ESNs leverage a relatively new model of neural networks called "reservoir computing". Reservoir Computing is a simple architecture that replaces the standard neural net with something more akin to a lake, in which several neurons flow and interconnect with each other. ESNs are very similar to something called a Liquid State Machine (LSM) (Schrauwen et al.). LSMs were developed independently from ESNs, and transform time-series data into activations on spiking neurons (Millea). ESNs are very similar in training procedure, but they use real valued neurons.

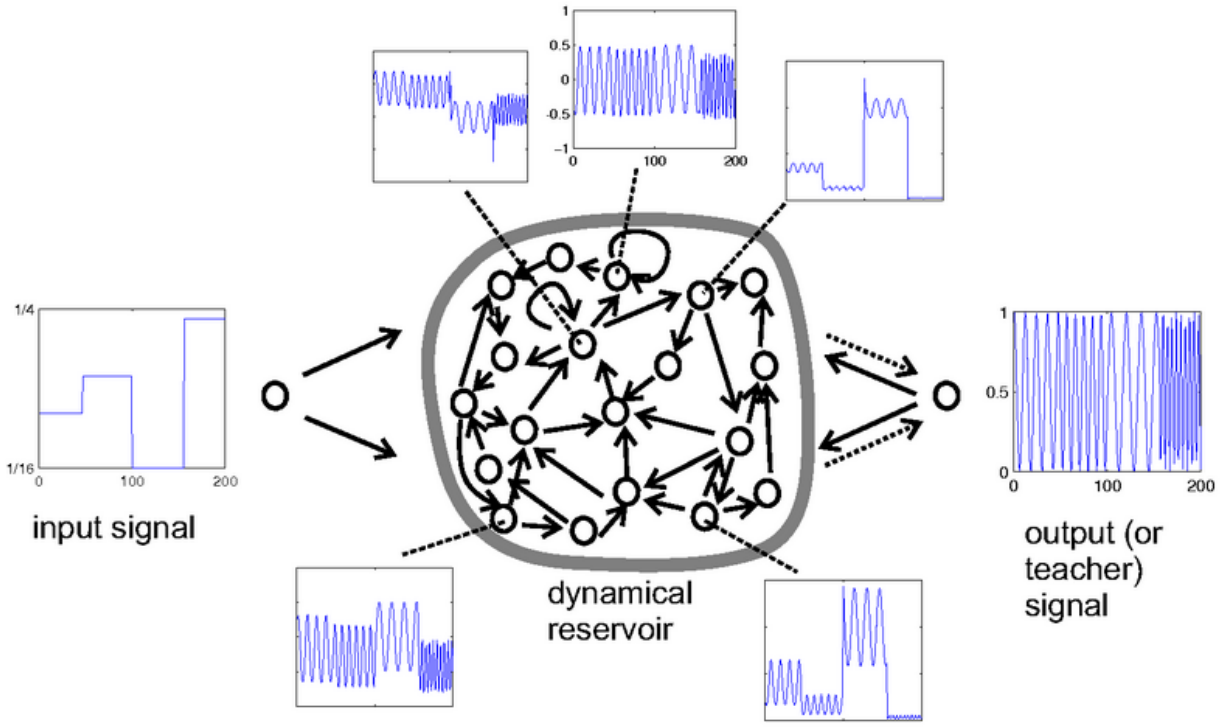

Figure 1. Inner Workings of an ESN. Image from Jaeger.

In figure 1, we can see the Inner Workings of an ESN. Each of the circles represent neurons. Unlike a traditional RNN, these neuron connections do not resemble a net, rather they look like a lake, or a reservoir. The ESN from this example is used to train a frequency generator. It takes an input signal from the input neuron (the circle to the very left of the image). Each one of the processing neurons in the dynamical reservoir contains a different weight, represented by the images connected by the dashed line. After the processing is complete, the signal is sent to the output neuron, where it is sent back into the reservoir for further processing. This system repeats until the number of necessary iterations have been reached. The dynamical reservoir is usually a completely random RNN, which explains its complexity and random weightage (Jaeger, 2007).

Equation 1: The main equation of an Echo State Network with inputs:

$$x(t + 1) = f(W^{in} \cdot u(t) + W \cdot x(t) + W^{fb} \cdot y(t))$$

Equation 2: The main learning equation of an Echo State Network:

$$W^{out} = pinv(M) * T$$

As seen in Equation 1, x(t) is the vector containing reservoir states at any time t. W^{in} is the connection between neuron i and n, while W represents the reservoir matrix. W^{fb} represents the feedback vector matrix, and y(t) is the output at any time t. The equation represents the driving phase of the network, when the output is cycled back into the equation as the input (Millea, 2014). In Equation 2, W^{out} represents the read-out vector, while T represents the desired output vector. The Equation represents a pseudo-inverse operation (Millea, 2014).

The ESN relies on something called the Echo State Property (ESP). The ESP relates asymptotic properties of the reservoir dynamics to the driving (input) signal (Jaeger, 2007). Essentially, the ESP will essentially just have the reservoir wash out any information from the initial conditions. The ESP is part of the reason the ESN can handle non seasonal, chaotic time series data so well. In one case, a team from Stanford created an ESN that performed even better than a Kalman Filter on predicting Google Stock Price data. The ESN had a test error of about 0.0027, while the Kalman error of about .2135 (Bernal et al., 2012). In that same study, the researchers tested ESN Performance on 50 randomly selected stocks from the S&P 500, and had median Variance Normalized Squared Error of less than 0.02 for all 50 stocks (Bernal et al., 2012).

Our ESN Model is based on a model used by Stewart. Stewart cites 7 major parameters of his ESN:

1. n_inputs (number of input dimensions)
2. n_outputs (number of output dimensions)
3. n_reservoir (number of reservoir, or processing, neurons)
4. random_state (seed for random generator)
5. sparsity (proportion recurrent neuron weights set to zero)
6. noise (regularization added to each neuron)

K-Means Clustering

K-Means is another traditional machine learning technique. It is normally used to cluster data points on scattergrams, and it can help identify relations and groups in data. K-Means clustering falls under a subset of machine learning called Unsupervised learning. Unsupervised learning means that the machine or system will essentially group, name, and label data without any sort of human intervention. K-Means clustering partitions data into k clusters, and uses euclidean distance to determine groups (Yıldırım, 2020).

The K-Means cluster is traditionally an iterative process. It randomly selects a centroid of a cluster, calculates distance, assigns data points, collects a mean of the data points to find new centroids, and then repeats the process until the cluster centroid stops moving (Yıldırım, 2020). The K-Means cluster is fast, scalable, and ultimately guarantees convergence. The distance algorithm used in a K-Means cluster is Euclidean, and follows this equation - $d = \sqrt{a\blacksquare^2 + b\blacksquare^2}$. K-Means clustering has been implemented in the Stock Market before. In 2010, a team from the Association for Information Systems Electronic Library (AISeL) used a combination of Hierarchical agglomerative and Recursive K-Means Clustering (HRK) to attempt to predict stock market prices (Anthony et al., 2010). They extracted quantitative and qualitative data from financial reports and stock quotes, and used Hierarchical Agglomerative Clustering (HAC), and Recursive K-means clustering to get representative feature vectors from the data. These vectors were then used to predict stock price movement Anthony et al., 2010). The team ran their framework on combination weight, and purity data, and reported an average accuracy in their predictions of about 60%, and average

profits ranging from 0% to 50% on combination weight data, and 20% to 40% on purity data (Anthony et al., 2010). The team also tested various types of clustering methods, and nearly all types of clustering methods maintained about a 60% accuracy (Anthony et al., 2010).

Methods

Our research aimed to verify the following hypothesis:

If we use a combination of Echo State Networks and K Means Clustering, we will be able to accurately identify a group of stocks that will maximize our short term (5 day) percent returns.

Unlike the studies which relied exclusively on K-Means clustering, we did not create vectors from the qualitative and quantitative data.

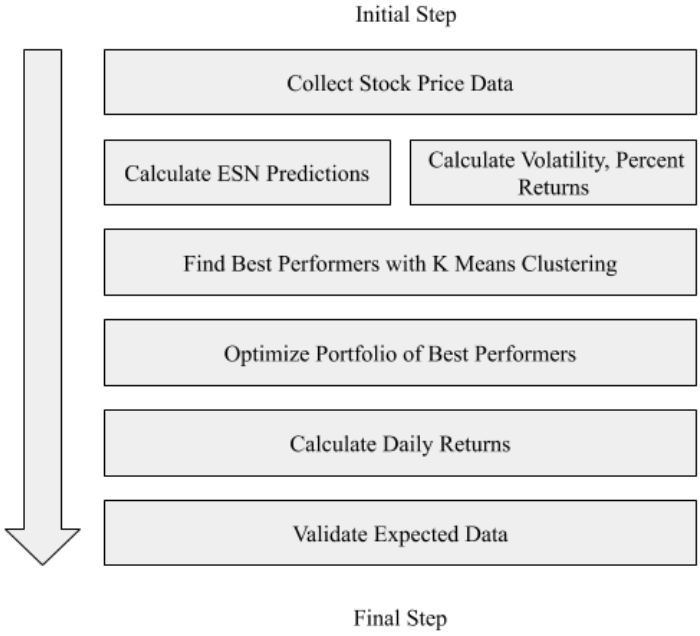

Figure 2. Framework used to approach our Research

As shown in Figure 2, we took a planned, multi-step process when developing our research framework. The initial goal was to collect clean, accurate Stock Price Data. We achieved this using the AlphaVantage Stock Price API. After Collecting our Stock Price Data, we split the data frame into two separate text files - one for returns, and one for share prices. We also had to reverse the files into order from earliest to latest.

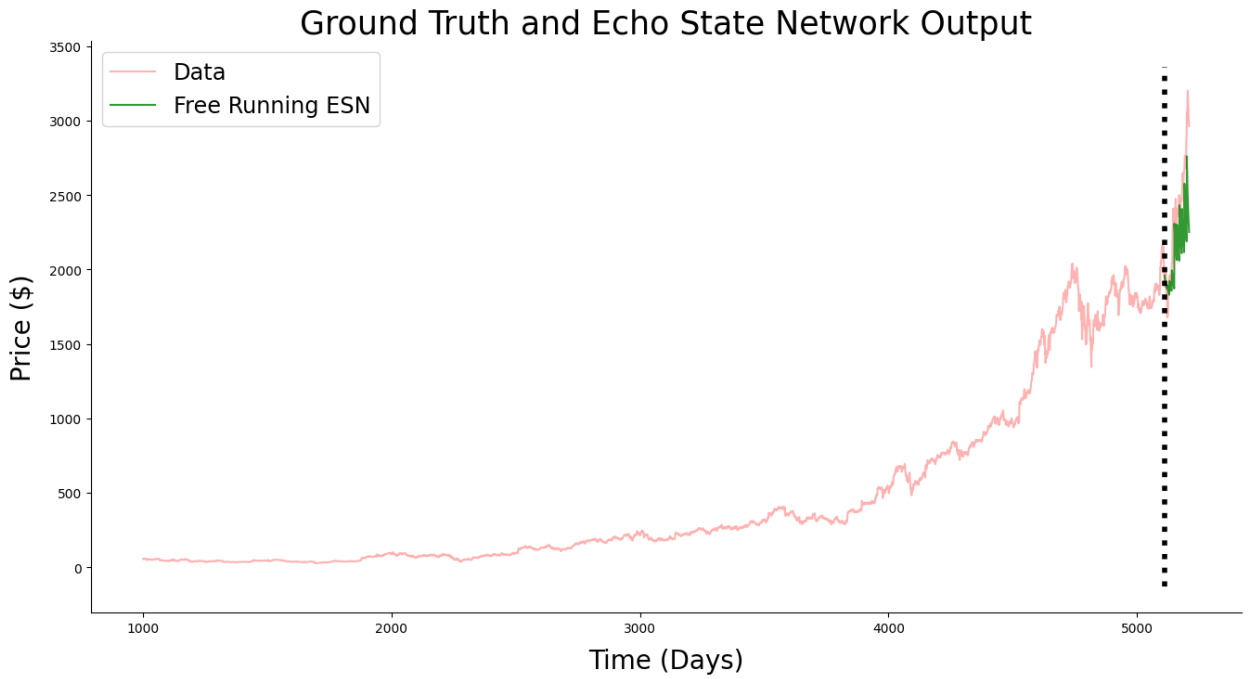

Figure 3. Example of ESN Output vs True Data on non-optimized hyperparameters with Amazon (AMZN) share prices.

As shown in Figure 3, the initial runs were based on arbitrary, non optimized hyperparameters. The approach taken to ESN Prediction was similar to the one used by Stewart. We did not follow Stewart's approach on Hyperparameter optimization, and optimized 0 parameters, not two. The reasoning being that Stewart focuses only on Amazon Share prices, and optimizing hyperparameters heavily on only one dataset can cause overfitting issues, especially when dealing with multiple datasets.

We selected 30 random stocks, and performed our ESN Predictions on them. We also recorded Mean Squared Error for each prediction. We let the model predict 100 days in advance from the 1500th day on market, and recorded actual percent returns and expected percent returns for each stock.

Equation 3: First Historical Volatility Equation used to calculate stock volatility:

$$r_i = ln(\frac{S_i}{S_i - 1}) \ \ for\ i = 0,1,2,3,\ldots,n$$

Equation 4: Second Historical Volatility Equation used to calculate stock volatility:

$$V = \sqrt{\frac{1}{n-1} * \sum_{i=1}^{n} (r_i - \underline{r})^2 * n}$$

In addition, we also calculated volatility using Equation 3. In the equation, r_i is the natural log of the stock prices at the end of each time interval (S_i), represented by time i, with (n+1) being the total number of observations. In Equation 4, $\underline{r}$ is the mean of r_i. Equation 4 calculates the standard deviation of r_i and multiplies it by the square root of the number of trading days in the year (n). We recorded volatility based on ESN Predictions, simulating how the algorithm

would "think" in a real world use case. We then made a scatterplot of our volatility vs percent return data collected from the ESN Predictions for each share. After running our recursive K-Means algorithm, we identified groups of high performers, and created a portfolio based off those clusters. We optimized our portfolio weightings by using the Monte Carlo Method to optimize returns.

Results

When running our ESN predictions, we took a step by step approach. We used data from the IPO until March 2nd, 2020 as the training data for each stock. Then, we ran predictions in 5 day steps. Each step was 5 days, so every cycle would predict the stock prices for the next 5 days, before verifying the data with the true dataset. We repeated this cycle 20 times for each stock, effectively collecting 100 business days worth of data. We chose March 2nd, as during the month of March, the Stock Market experienced a major recession brought about by the Coronavirus outbreak, and we wanted to see how our model performed with that knowledge.

We recorded the average percent returns every 5 days for each ESN. We compared that average to the true average percent returns every 5 days for the real data.

Table 1. Random Selection of 30 Stocks chosen for Testing, the Mean Standard Error (MSE) between the ESN and the True Data Prediction, Average Percent Return (ESN), True Average Percent Return, and the Volatility score for each Stock. Spectral Radius was 1.1, with a noise of 0.0005.

Share	MSE	Volatility	Avg. 5 Day Percent Return (ESN)	True Avg. 5 Day Percent Return
AAPL	15.7	0.63192	1.575%	1.552%
ALACU	.324	0.1474	0.3113%	0.0482%
AMH	1.372	0.2234	0.0045%	0.2751%
AMZN	311.647	0.5175	9.693%	2.178%
ATKR	26.4968	0.54329	-19.13%	1.622%
BKCC	46.659	0.4951	-173.4%	-0.500%
CABO	120.575	0.2709	0.6078%	0.9158%
CIGI	6.4264	0.3949	-0.6353%	-0.7815%
CNBS	91.321	0.4822	52.73%	0.7951%
CPAA	0.542	0.2316	0.8962%	0.3022%
ETP	1.229	0.4314	1.165%	0.6545%
EWH	0.779	0.2634	-0.1559%	-0.1923%
FDNI	4.000	0.3000	3.495%	2.127%

FMN	0.599	0.1711	-0.0263%	0.0213%
FNK	2.211	0.2323	0.0234%	-0.1252%
FNLC	1.7815	0.4493	-0.9769%	-0.8543%
FUT	0.342	0.0508	0.0762%	0.0828%
GOOGL	62.546	0.3493	0.2667%	0.5368%
HBANO	11.291	0.1617	5.253%	0.1273%
IBM	6.184	0.2674	-0.8271%	-0.4500%
IFRA	225.5	0.2793	1546%	-0.243%
MIY	0.4298	0.1322	-0.0991%	-0.0932%
MLR	1.4124	0.5999	-0.2219%	-0.1494%
NAK	0.1682	0.7998	30.82%	7.773%
PROV	1.2967	0.3989	-2.253%	-2.139%
RDIV	2.2581	0.2156	0.5590%	-0.6656%
TPIF	1.5374	0.3585	0.1510%	0.1084%
TSLA	88.723	0.5474	4.056%	4.232%
TSM	2.5104	0.4302	0.8687%	1.171%
ULTR	0.5306	0.0292	-0.0782%	-0.0666%

Table 1 represents the raw data we collected from our models performance. In most cases, the model seems to be accurate, although sometimes it can vastly overestimate trends, again reflecting the possible volatility of the stock.

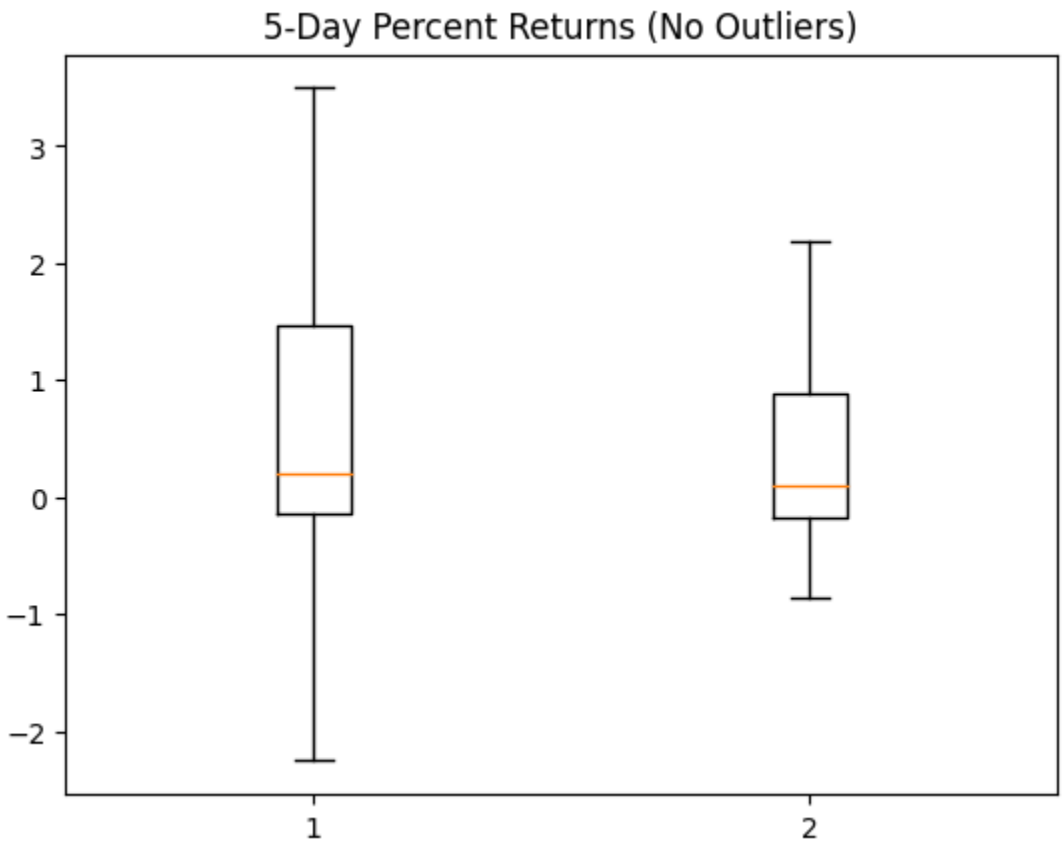

Figure 4. Average 5 Day Percent Returns from Table 1 spread onto a boxplot. This figure does not include outliers. Trace 1 is the ESN Data, while Trace 2 is the True Data

If we look at Figure 4, we can see that the spread of the ESN Data (Trace 1) is larger than the True Data (Trace 2). However, the median spread on both traces is around the same (close to 0), showing that while the ESN can skew high or low when faced with sharp upturns and downturns, in the long run, it can semi-accurately predict trends and share prices.

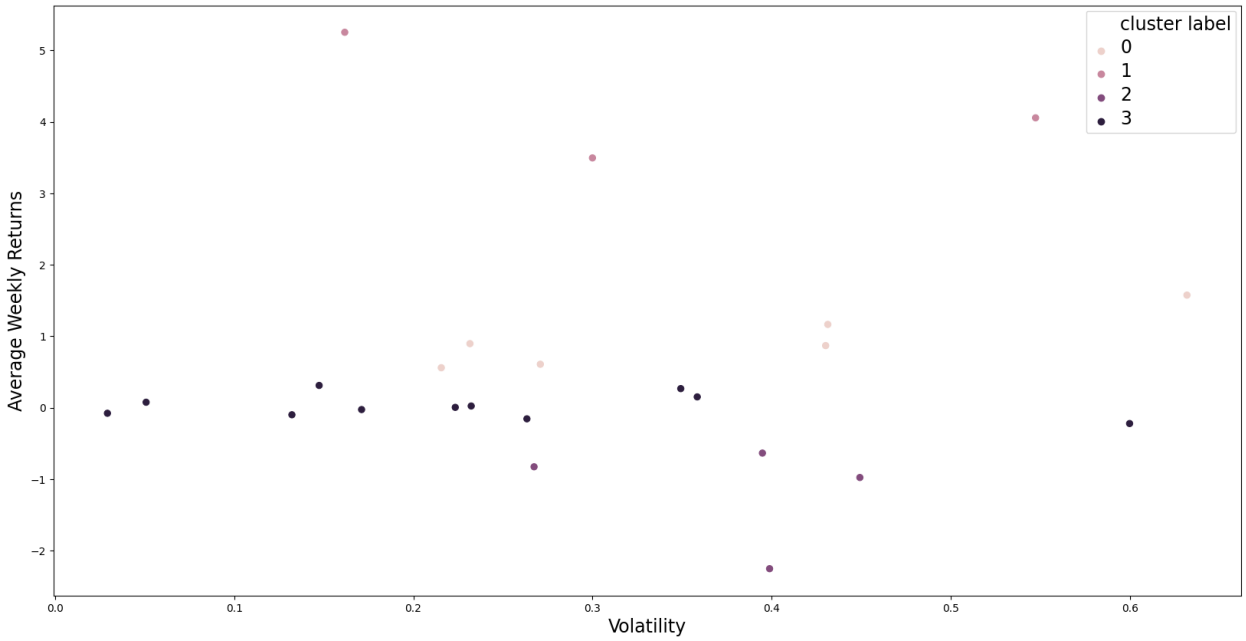

Figure 5. K-Means Cluster with 4 Cluster groups of ESN Data. Spread of Average 5 Day returns vs Volatility.

Figure 5 outlines the initial clustering of our data. The K-Means cluster was arbitrarily given 4 groups to place into. While Volatility spread was very high, the returns spread can be classified into distinct groups. High Returns does not necessarily mean high volatility, as one of the predicted high performers had one of the lower volatility measurements.

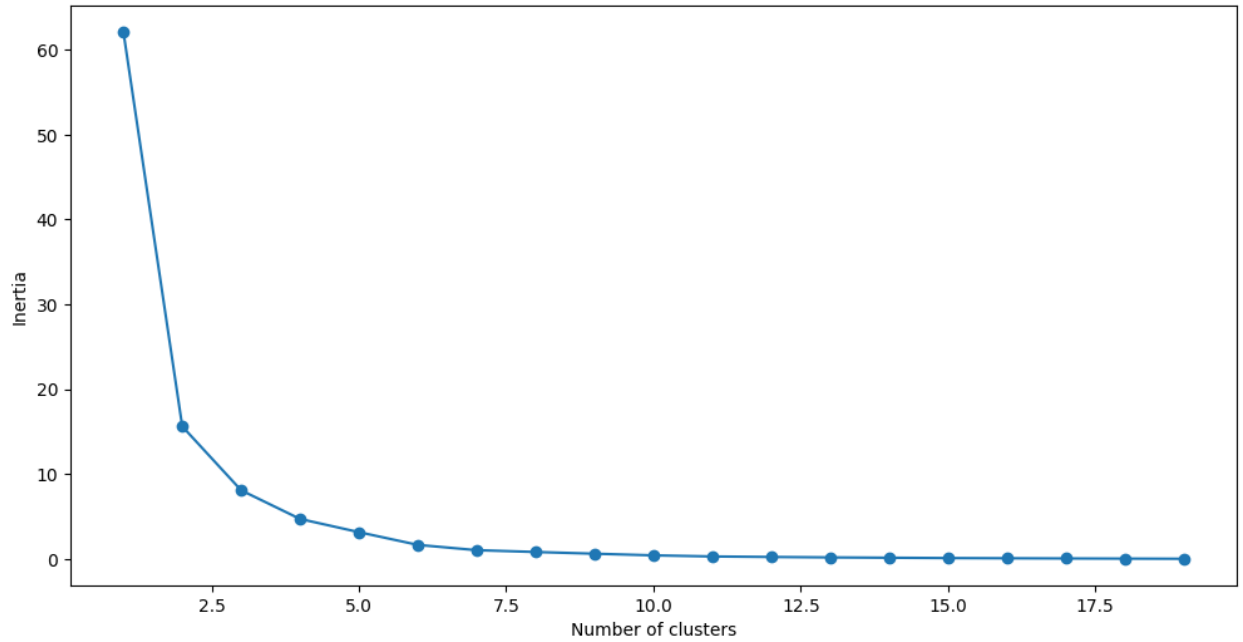

Figure 6. Elbow Graph outlining cluster vs. Inertia spread for our dataset

As shown in Figure 6, K-Means clustering will have diminishing returns (inertia approaches 0) as the number of clusters increases. So, common practice is to select the Elbow point, or in our case, about 4 clusters. Thus, our arbitrarily chosen value will work.

Since Volatility risks are about the same across all groups, we will select cluster group 1, which are the highest performers. Our highest performers were FDNI, HBANO, and TSLA. Now we enter our final step which is portfolio optimization based on expected returns.

After grouping our stocks, we decided to test our algorithm. The dataset contained values up until July 20th, 2020. So we decided to see how our algorithm would perform based on real world data. Our ESN Predicts report average returns of about 5.23% for HBANO, 4.06% for TSLA, and 3.49% for FDNI. Comparing our cluster predictions to real world numbers, our True Averages for HBANO, FDNI, and TSLA, were about .127%, 2.127%, and 4.232% respectively.

Discussion

Our Data Collection shows that after accounting for outliers, ESN's can predict stock trends with varying degrees of accuracy. Our ESN reported 4 cases of when ESN predicted an increase, and the true value reported a decrease (or vice versa), and 3 of those cases were outliers. We achieved 96% accuracy when predicted trends across our averages, but our ESN was not nearly as accurate when predicted how much the stock would increase. Our K-Means clustering groups also were relatively accurate, with the only large difference between predicted stock and true stock was HBANO shares, with predicted returns being 41.18 times larger than the true gains. Only TSLA shares outperformed the predicted gain by a slim margin, about 1.04 times larger than the predicted returns.

Conclusion

In lieu of our findings, we propose a possible model for future algorithms. We propose using an ESN trained on all historical data of 100 sampled shares to predict the returns and share prices for the next week. We then propose calculating volatility based on these predictions, and then clustering the middle 50% of stock data with K-Means clustering (to avoid possible skewing by outliers). After identifying the optimal portfolio, the algorithm would run Monte Carlo testing to optimize the portfolio weightages of stocks, and theoretically maximizing returns. At the end of the close at the end of the week, the algorithm would sell all shares, and repeat the process. In all, we concluded that while our ESN could accurately predict stock trends, relying only on historical data was not enough to predict returns at the accuracy level (95%+) needed to have a successful algorithm trading bot.

Limitations

Due to time constraints, and the inability to find a proper backtesting platform for ESN based algorithms, we were unable to determine the historical performance of this algorithm. In addition, the sheer amount of training data needed to generate semi-accurate results from our ESN meant that we could not build, test, and train our algorithm and analyze real world performance, as we used data up to July 20th, 2020 for training and testing purposes. Finally, limitations from 0% commission trading APIs and SDKs, and lack of necessary computing power meant that we could not create a bot that would be fast enough, or have the necessary software to perform algorithm trading. One of the major limitations of our research was on the ESN itself. Stewart reported lower accuracy when predicted past 1-2 days, something we did see with our 5 day prediction time window. However, our algorithms process of selling everything every week meant that it would not be practical to implement daily, and could result in serious losses in a volatile market with large daily swings up and down.

Acknowledgements

We would like to thank Thomas Jefferson High School for Science and Technology for inspiring me to pursue this research. We would also like to thank PHD Student Adrian Millea for publishing his Master's Thesis, and Dr. Matthew Stewart for publishing his article as they provided the basis for some of the key concepts and research on Echo State Networks necessary for this paper.

References

Anthony, J., Lin, M.-C., Kao, R.-T., & Kuo-Tay, C. (2010). *An Effective Clustering Approach to Stock Market Prediction. 54*. https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1030&context=pacis2010

Bernal, A., Fok, S., & Pidaparthi, R. (2012). *Financial Market Time Series Prediction with Recurrent Neural Networks*. http://cs229.stanford.edu/proj2012/BernalFokPidaparthi-FinancialMarketTimeSeriesPredictionwithRecurrentNeural.pdf

Grossberg, S. (2013). Recurrent neural networks. *Scholarpedia*, *8*(2), 1888. https://doi.org/10.4249/scholarpedia.1888

Jaeger, H. (2007). Echo state network. *Scholarpedia*, *2*(9), 2330. https://doi.org/10.4249/scholarpedia.2330

Kim, T., & Kim, H. Y. (2019). Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data. *PLOS ONE*, *14*(2), e0212320. https://doi.org/10.1371/journal.pone.0212320

Millea, A. (2014, June). *Explorations in Echo State Networks*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.722.4917&rep=rep1&type=pdf

Schrauwen, B., Verstraeten, D., & Campenhout, J. (n.d.). *An overview of reservoir computing: theory, applications and implementations*. Retrieved July 22, 2020, from https://biblio.ugent.be/publication/416607/file/447949

Stewart, M. (2020, July 18). *Predicting Stock Prices with Echo State Networks*. Medium. https://towardsdatascience.com/predicting-stock-prices-with-echo-state-networks-f910809d23d4

Total Market Value of U.S. Stock Market. (2020, July 3). Siblis Research. https://www.google.com/url?q=https://siblisresearch.com/data/us-stock-market-value/&sa=D&ust=1595447784312000&usg=AFQjCNHI_egTg096YQbTSuEMARwPdFnWpg

What Is Backpropagation? (2017, December 7). Edureka. https://www.edureka.co/blog/backpropagation/

What Percentage of Americans Owns Stock? (2019, September 13). Gallup.Com. https://www.google.com/url?q=https://news.gallup.com/poll/266807/percentage-americans-owns-stock.aspx&sa=D&ust=1595447784311000&usg=AFQjCNGrZSWIKPio3NFEznOHAKyQEJaJRg

Yıldırım, S. (2020, March 3). *K-Means Clustering — Explained*. Medium. https://towardsdatascience.com/k-means-clustering-explained-4528df86a120