CSCI 6502 - Big Data Analytics
Big Data Analytics for Portfolio Management

Aser Garcia
asga4869@colorado.edu

Abstract

Allocating and managing a portfolio of financial assets is important to any person that wishes to retire or increase their level of wealth. Portfolio management in asset allocation and re-balancing is costly if done by a professional or has sub-par performance if done by a computer algorithm. In this project, an intuitive dashboard is created to showcase the advice from different algorithms and including those with statistical and machine learning support. This will be achieved with big data and recurrent neural networks measured against traditional methods of portfolio rebalancing.

1 Introduction

Allocation and management of assets has been around since humans could own property, trade merchandise, and currency was invented. The true problem is allocating the right assets with minimum upfront cost, high returns, and considering the investor's preferences on risk. In modern day markets there are mathematical frameworks to decide on the structure of your assets, other wise known as portfolio allocation. Harry Markowitz, a Nobel Prize winner in economics, presented the "Modern Portfolio Theory" through his dissertation in 1952. In this essay he details that portfolios should consider the variance of each asset towards the whole allocation of assets for risk. Minimizing risk and maximizing profit is the high-level idea of the framework. Today this strategy is regarded as fundamental in the allocation of assets.

Several services offer the public ways to create and maintain a portfolio by either the use of professionals in the finance industry or algorithms giving the advise known as "robo-advisors". In the past, people have trusted their human advisers to provide adequate information and provide the highest return of investment. With millennials reaching adulthood studies have shown that 66% of children who receive an inheritance fire their parent's financial adviser shortly after[5]. The assumption is that millennials are more used to automation of tasks leading portfolio optimization to be no different. For this reason, there is a 500 billion dollar market estimation for 2020 in the "Digital Advice Market".[5]

In this work, we explore a system to collect daily data while setting measures for fault tolerance and scalability. In addition, a machine learning model with layered gated recurrent units (GRUs) for portfolio management is compared against regular statistical optimization over a ten week period.

2 Related Work

Big Data in Portfolio Allocation: A New Approach to Successful Portfolio Allocation by Irene Aldridge[1] considers the inverse of the correlation matrix as the most significant factor in re-balancing a portfolio. Aldridge shows that the inverse correlation matrix is much more informative than the stand alone correlation matrix. To maximize return the author uses principal component analysis (PCA) on the inverse eigen-vectors and eigen-values to determine which assets to keep and which to adjust after each month. These eigen-values and vectors caputure the most variablility in the data thus the pool of assets to consider can be greatly reduced. The results show a 400% increase in return compared to other methods such as "vanilla" mean-variance optimization and PCA on just the correlation matrix.

In "An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown" by Almahdi et al[2] train a recurrent

reinforcement learning (RRL) model to adjust portfolio assets based on the weights and market trends. Almahdi et al. show that their RRL model outperforms their baseline buy-and-hold portfolio. In addition, it is noted that the RRL model does better to decipher the market noise from trading signals.

Harry Markowitz [7] in his 1952 dissertation set the groundwork for modern portfolio theory. Here he mathematically describes the process of portfolio selection through the variation of each asset considered. In the paper Markowitz describes the process as being the second stage of portfolio selection. The first stage would be the observation of historical and current data.

Moore et al. [8] in their paper Mathematical Models in Portfolio Selection list several models for the selection of assets to allocate. This is more of a survey paper highlighting the work done by Markowitz, Sharpe, and others up to the publication in 1972. At the end of the paper they show computer simulations to show the performance of the models.

3 Methods

3.1 Dataset - Alpha Vantage API

Data is stored for the companies listed in the Standard and Poor's five-hundred (SP500) index as of this year of 2020. Henry Varunum Poor founded the company "Poor's Publishing" and published an investor's guide during the railroad industry's boom in mid to late 1800s. In the early 20th century the Standard Statistics Bureau, later renamed the Standard Statistics Company, developed the first stock market index. By the mid 20th century the two companies merged to form the Standard and Poor's Global, or just SP Global.

The selection criteria to be included in the index is a company must have a market capitalization, or the total value of the company in stocks, of at least 8.2 billion dollars. The annual dollar value traded to available company stocks is greater than 1.0. Lastly, the minimum monthly trading volume is of 250,000 shares in each of the six months prior to evaluation.

The SP500 index was chosen because the companies listed are less likely to have high variability in their stock prices. Less variability translate to "safer" investments. Along with less risk, the index is comprised of the top five-hundred companies with the highest impact in the market. Only companies based in America are included in the index.

Data is being gathered using the Alpha Vantage API with a Python wrapper. The advantage of this API is that it is free to obtain historical data and it is accessible with Python. The data provided by Alpha Vantage is historical enough to be relevant. At latest, the data is of the early 2000s years which would be relevant to today's stock for that company. This historical data includes the daily open, high, low, closing price, and volume of the stock from the first day the business went public to the present day.

The data was gathered on a daily interval. First there was a collection of historical data for each company to populate the tables in the database. Afterward, the API calls are scheduled for two hours after the New York Stock Exchange closes that day. The calls execute every fifteen seconds at a maximum of five-hundred calls a day. The throttling of the API calls keep us in the bounds of the free use on the Alpha Vantage API, which is five calls a minute capped at five-hundred calls a day.

Going over the attributes of the data, the data is structured since it comes from a table that contains the information for that day on a particular stock. In regards to the characteristics, or five "V"s of big data, the volume is both low and high. I consider it high volume since we need to collect the historical data first and store. It is of low volume since we are only getting five data points per stock at a maximum of five-hundred stocks. It is low velocity because we are updating all the tables once a day. It is high veracity because the stock exchange market must be accurate. Lastly, the value of the data is high-value because we can train deep neural networks on economic recessions and other disasters.

As of March 30,2020, the collected data summed to 2,284,338 data points across the 500 companies in the SP500 index.

3.2 Stream Processing - Kafka

Stream processing was necessary for this project due to data surfacing every work day. The issue was the need for scalability on the streaming since the focus was to make the project as adaptive as possible. Currently, the work deals with portfolio management and re-balancing a portfolio every month. It would be prudent to set tools for scalability in the event of transitioning to higher frequency trading. For this reason, Kafka

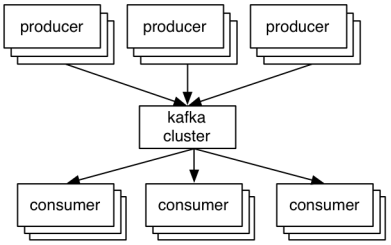

Figure 1: Kafka cluster setup and use[6]

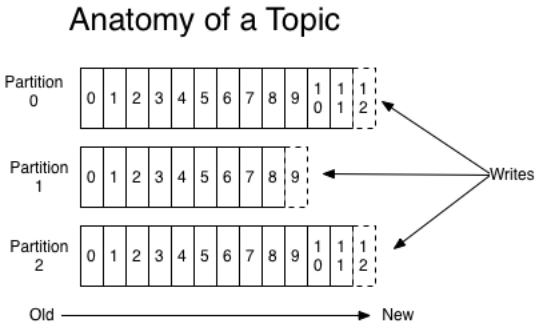

Figure 2: time scale of different topics partitioned in a cluster(provided by kafka.apache.org)

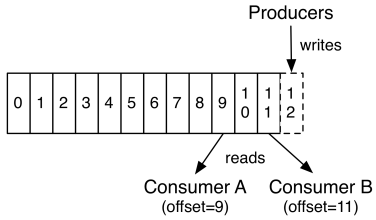

Figure 3: producers writing to a topic log, consumers reading at different offsets for any partition (provided by kafka.apache.org)

streaming was used to handle stream processing.

Kafka was developed by LinkedIn, the professional employer/employee social networking site, with the purpose of being a scalable, fault tolerant, message subscription distributed streaming platform. Kafka later joined the Apache Software Foundation. The distributed part means that Kafka can be run as a cluster on multiple servers if needed. It has multiple APIs for producing streams, subscribing to multiple streaming topics at once, processing streams, and reusing the producers and consumers, and finally the admin API which allows for managing and inspecting topics.

To dive a little further on how Kafka leverages topics and records in order to be a little fault tolerant we will analyze figures 1-3.

In this image we see three partitions with different amounts of writes. Having an immutable record allows to use previously written data in order to analyze. There is a time window that can be set in order to reduce the memory usage of the cluster on the machine. This allows for multiple consumers to subscribe to a topic and have a lag, or offset, to read previous data. This log of records way of managing streams makes it fault tolerant, scalable, and easy to use.

Kafka is not only quick to configure and fault tolerant, it is also highly scalable. Kafka has two major components to stream data. The first is the Kafka producer, this is the worker that handles the sending of data and keeping track of a log. The second is the Kafka consumer which catches the data sent and can listen to multiple topics of data at once in order to process accordingly. The Kafka cluster is the mediator that handles the logs of records for each topic. With the quick development of the Python language, Kafka producers and consumers were configured to catch data as it was being requested through the API.

A Kafka producer was created attached to the Alpha Vantage API in order to stream the data to the Kafka consumer. The consumer then caught the data and handed it to the database for storage. The key here is the future of this project can certainly adapt to a higher frequency of trading due to Kafka's scalability and fault tolerance which is extremely important in high frequency trading. Jay Kreps[6] wrote an article for LinkedIn reviewing and stress testing Kafka. For the future purposes of this project, Kreps found that the end-to-end latency from producer to consumer is 2 milliseconds at the median with a high 99.9th percentile of 14 milliseconds. This is very useful for future high frequency trading endeavors.

3.3 Database - Cassandra

Continuing with the idea of scalability and fault tolerance, the Cassandra database system was used for storing the data gathered from Kafka streams. Cassandra was designed to be easily scalable, whether that was in adding more nodes to a cluster or a completely new cluster. Several copies exist for the data in the cluster which is necessary for the vital information in historical prices to further process later.

Stress tests were administered by the Netflix team in a blog by Adrian Cockcroft and Denis Sheahan[3]. During these tests they were able to

show 1.1 million writes per second to Cassandra using 60 instances of clients. I am only using one client instance which greatly increases my writing capabilities as well as for future writing.

Python was used to connect and configure the cluster initialized on my local machine. This can be transferred to a stand alone machine and then add more nodes to that machine in order to create cluster. This is called linear scalability and each machine performs the same. Read and write throughput also increases linearly with the addition on new machines to the cluster.

Since this is a NoSQL type of database, we are trading the SQL capabilities of data manipulation for speed and scalability. Currently the Cassandra cluster on my local machine contains all 500 needed tables to form the SP500 index of companies. Each company has data points from the beginning of when they became a publicly traded company to the present day. This adds up to a bit over 2.2 million data points.

With Kafka and Cassandra combined I was only limited to the speed limit of Alpha Vantage. For the future, also mentioned in the future works section, I will transfer data producing from Alpha Vantage to Quandl which is another company that provides stock time series data. The difference is the availability for more calls to their API after recieving a free key. After switching to Quandl, the new speed limit would be 500 calls per minute, at a limit of 720,000 calls per day greatly expanding our stock selection criteria and the speed of data collection.

3.4 Mathematical Frameworks

Having baseline mathematical frameworks was vital for comparing the machine learning model in this project. As we mentioned before the fundamental strategies to asset allocation was the use of mean-variance optimization developed by Harry Markowitz.

The problem presented by Harry Markowitz is an optimization problem. Suppose you had a target return for a portfolio, represent that as $\bar{\mu}_{target}$. For each asset in your portfolio, assign a certain weight to distribute your investment into. This weight is the amount of the investment that goes into a particular asset, we represent it as w. Assuming that we cannot borrow stocks, our portfolio must sum to 1.0 and each weight is greater than zero. Now define a variance-covariance matrix Σ for the stocks in question. Each stock has a mean of their returns, call that μ which is a vector where each entry is the mean of a particular stock. Our optimization problem is as follows, for M stocks:

$$minimize_w \quad w^T \Sigma w$$

subject to

$$w^T \mu = \mu_{\bar{target}}$$

and

$$\sum_{i=1}^{M} w = 1$$

and

$$w \geq 0$$

This was solved in the project using Python's Scipy library that has an optimization algorithm to optimize an equation based on the constraints provided.

A different strategy of portfolio allocation and management was presented by William F. Sharpe who won the Nobel Prize for the Sharpe Ratio. The major difference is that although Harry Markowitz's optimization finds the minimum risk for a given target return, the Sharpe ratio tries to maximize the portfolio based on every unit of risk. This means that the portfolio becomes more risky but the risk is justified due to higher returns. Keeping the same representations for the portfolio weights, variance-covariance matrix, and the mean returns of each stock our optimization problem is now, for M stocks:

$$maximize_w \quad \frac{w^T \mu}{\sqrt{w^T \Sigma w}}$$

subject to

$$\sum_{i=1}^{M} w = 1$$

and

$$w \geq 0$$

This is known as greedy optimization since we are trying to maximize our returns adjusting for risk. This optimization problem was also solved numerically using Python's Scipy library.

The above mentioned optimization by the Scipy library will take in as argument an evenly distributed portfolio at first, the variance-covariance

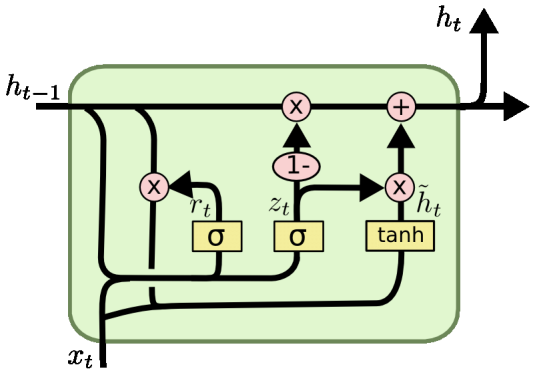

Figure 4: Internal Structure of a Gated Recurrent Unit.

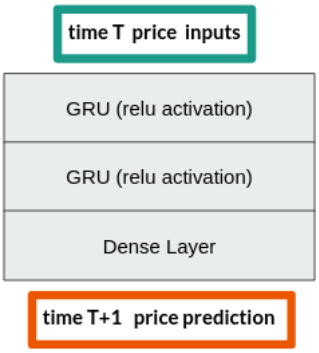

Figure 5: Model architecture to predict stock prices over a portfolio of stocks.

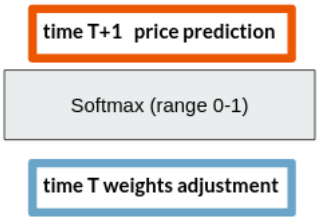

Figure 6: Added step to portfolio managing with machine learning GRUs

matrix, and the average returns vector and then return a weights matrix that is optimal given the constraints. This optimization algorithm is executed every time we need to rebalance our portfolio.

4 Second Stage Methods

4.1 Models

Apart from the traditional mathematical way of choosing assets for a portfolio, machine learning models will be used. Recurrent neural networks have shown promise in the allocation and management of portfolio stocks by Angelos Filos[4]. Dr. Filos proposed a specific architecture for stock allocation using a type of recurrent neural network, the gated recurrent unit. The internal structure of a GRU is depicted in Figure 4. Here we can see the simplicity of the internal structure compared to a traditional RNN or LSTM. The reason for using a GRU versus any other type of recurrent neural network is that no only does it take care of the vanishing or exploding gradient of a RNN but also the time cost for training an LSTM. Since it does not have three outputs like the LSTM it had less gradients to compute when performing back propagation over time.

The architecture implemented in this project is shown in Figure 5. We have two stacked GRUs followed by a fully connected layer to output predictions of the next price of a stock. Our input vector is a vector of size M with stock prices for each stock at a time T. The model then returns a vector of size M with predictions of the prices for the next time step, T+1. Since we want to adjust our portfolio to weights of the stock we include a softmax layer. A softmax layer is key to portfolio management with a GRU since it will return a vector of size M with probabilities proportional to the sizes of the predictions. If the prediction is high for a stock then the weight for that stock produced by softmax will also be portioned high, keeping the information from the prediction without having to be very accurate. A stacked layer of GRUs was chosen because of the complexity that is added to the stock.

Tensorflow 2.0 was used to create the models and the fitting and prediction functionality of the models. Tensorflow is a library and framework for developing and training machine leanring models with either the language C++ or Python. I decided to go with Tensorflow because of the available support for the library, the large amount of tutorials to learn how to use tensorflow, and the ease to connect the library to a GPU. Keras, a machine learning library specifically for neural networks, is now part of tensorflow which was used in order to build models, train the network, and provide predictions. Python was used for the development of the model and a local NVIDIA GTX 1060 Mobile GPU was used for training models and prediction. The local GPU really reduced the training time for the models since the GPU was not shared and has all the available memory possible for training. Even the tensor processing unit, a piece of hardware specifically designed for matrix multiplication, was slower than the local GPU mentioned. This allowed me to train

models for each type of portfolio a lot faster than using Google Colaboratory. In addition, training locally allowed me to save the model without jumping through the hoops of saving to Google Drive or any other type of online storage system.

4.2 Visualizations

For the visualization portion of the project, I decided to use a library called Plotly. Plotly allows for gorgeous graphs and interactive environments for graphs that can be run locally on the browser or through their Dash framework which creates a local host to test on for deploying to a server later. There were major advantages to using Plotly for this project. Plotly supports many data frame packages which includes pandas, a very popular data science tool for data manipulation and analytics in python. Without too much effort I was able to make sliders for time series data as well as pie charts that looked great and development time was kept to a minimum. In addition, the plan was to create a front-end that allowed the user of this system to visualize the portfolios that were suggested by both the mathematical frameworks and the GRU network. This was easily done with Dash, an open source package for Python that requires no additional learning to be able to make great web pages with all the necessary plots. By passing the graphs to a Dash app object, the package took care of the HTML, CSS, and Javascript needed for graph placement, colors, formatting, and interaction with the graphs. There was no extra work to be done.

5 Results

5.1 Assumptions

Before communicating the evaluation for the models there were several assumptions that had to be made about the trading environment in order to achieve preliminary results in this project.

1. Insignificant Market Impact: Since we are training and testing on data and prices of stocks that has already happened, we will have to assume that our investments do not affect the market in any way. Since we are working with the SP500 dataset, we know that the requirement to be listed in the index is having an available stock value of 8.2 billion dollars. Investing a couple thousand will not affect the overall trend of the company. Thus we can assume that our choices in stock selection have no market impact.

2. No transaction costs for trading stocks: In the real world, there are brokerages that require a fee each time an individual trades a stock. This means that our returns are reduced a certain percentage after every rebalancing. There are some brokerages, especially automated brokerages that do not need humans to place trading requests and thus do not need a fee for trading. In this project we assume there are no trading fees.

3. All assets have a high liquidity: A liquid asset is an asset that can be traded for money, sold or bought, without any delay. A non-liquid asset would be real estate or automobiles. Since we are adjusting our portfolios and calculating returns really quickly through a computer then we assume that the returns are the true returns since the stocks could be traded or bought immediately.

5.2 Initial Asset Allocation

After setting these assumptions we are confronted with the problem on how to choose a stock with the variety of sectors and stocks to choose from for each sector. In this project we utilize the statistical forms of allocation to make a first allocation and then manage the portfolio using the machine learning models and the statistical strategies.
The strategy used for initial allocation is known as ”top down” investing. The term ”top down” refers to analyzing the market in larger parts and then centering on the smaller parts. For example, in this project, the average stock prices for all the companies in a certain sector, such as health care, for the SP500 companies was made into a time series. This was done for all the available sectors in the SP500 which was composed of 11 sectors including materials, utilities, consumer staples, real estate, etc. The portfolio on each strategy in shown in Figure 7.
After running the optimizations algorithms for a minimum risk allocation and a Sharpe ratio allocation we could could see the optimal sectors to invest in. Afterwards, time series data of the stocks for each sector was collected through the database and then analyzed. For each sector we calculated a new portfolio and then multiplied the weights in that portfolio by the weight of the sector that was previously found. Any weight that was zero for a sector or a stock in a sector was removed. The rule of thumb is for a portfolio to have more than 25

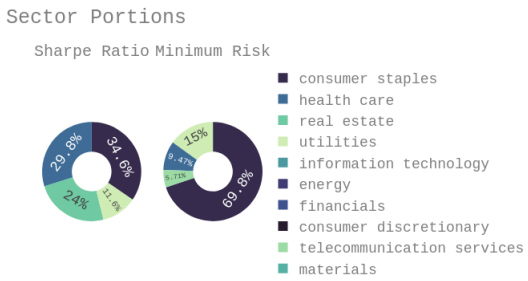

Figure 7: Top Down Investing: Sector Analysis (Plotly dashboard)

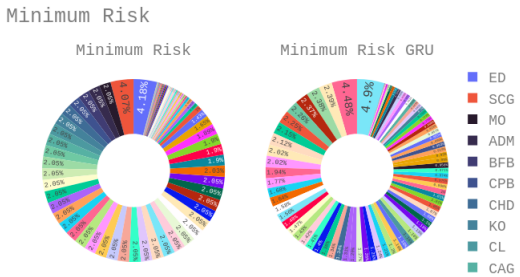

Figure 8: Top Down Investing: Sharpe Ratio allocation from statistical methods and GRU model. (Plotly dashboard)

different stocks. If transaction costs are negligible then upwards of 50 stocks is recommended[9]. For the GRU network, the initial weights are even for all the stocks chosen by the optimization of both portfolio strategies and then the model is trained on the history of the stocks. After the model is trained it tried to predict the next time series stock price and applies the softmax activation to get the weights best suited for returns to that prediction. Standard prediction is used in this context without normalizing the data not applying different functions like log. What we get is two different portfolio allocation based on the statistical methods and the GRU network. shown in Figures 8 and 9.

5.3 Allocation Strategy Comparison

In addition to initial allocation, testing the GRU model against the statistical methods was necessary. For that a plan was devised. Gathering the data from these stock choices of the initial allocations we split that data into a training, validation, and test set. The test set was split by the ending

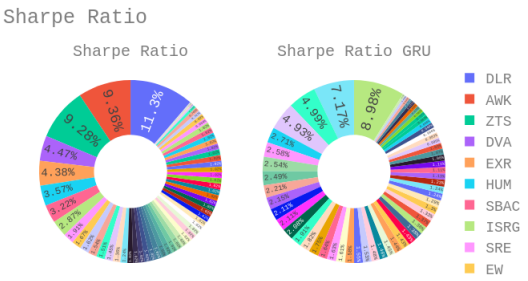

Figure 9: Top Down Investing: Minimum risk allocation from statistical methods and GRU model. (Plotly for graphics)

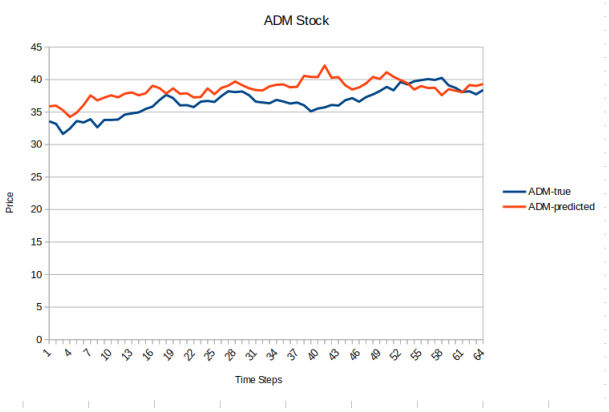

Figure 10: Sample true and predictions of ADM stock on validation data.

ten weeks of the time series. This happened to be the beginning of January 2020 to mid March 2020. This was strategically chosen because the market for the SP500 stocks entered a failing economy during this period due to the pandemic caused by COVID-19. To refrain from going into too much detail, most of the country was required to be under quarantine to reduce the spread of the virus. Businesses did not have as much traffic as usual so the economy went into a poor state. Similar to the 2008 recession with a much steeper drop, I wanted to test the models under these conditions. The rest of the data was used for training and validating the GRU models.

We can see results of prediction in figures 10 and 11 for different stocks in the Sharpe ratio strategy portfolio. After training the models we set up a loop to consider each week in the testing set. Both models started with even weights and adjusted the weights after each week. The statistical optimization methods calculated the optimal portfolio on

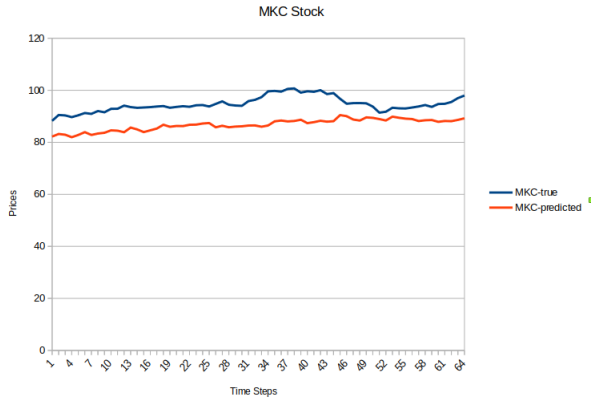

Figure 11: Sample true and predictions of MKC stock on validation data.

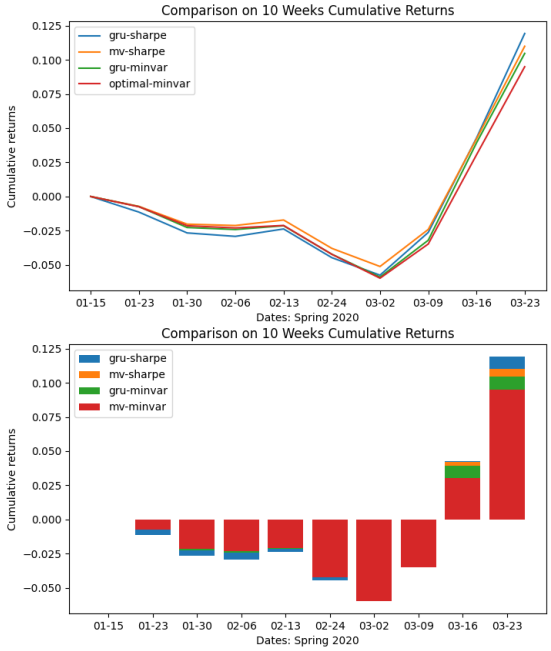

Figure 12: GRU and Statistical Methods of portfolio management over a 10 week period at the beginning of the year 2020

the past week to adjust for the upcoming week. The GRUs took in the past week and tried to predict the future prices in order to make adjustments with softmax activation. Percentage change was calculated on the previous week including the following day in order to calculate the returns of the rebalancing. Each week's return was added together to track a cumulative return in order to see the accumulation of wealth overall by the adjustments which is not time step specific. In Figure 10 we see the results over the 10 week period.

Since the stock market took a dive at the beginning and middle of the pandemic, not only from companies that relied on China for materials and goods but also locally in the United States of America both strategies for portfolio management did poorly and then rose again after the government tried to help the economy through loans. The clear thing to see is that stocks allocated using the Sharpe ratio are more volatile and thus strategies managing those stocks can have really poor returns or really great returns. The stocks chosen by the minimum risk allocation strategy remain closer to zero since they are less likely to fluctuate, therefore leading to less positive or negative returns.

6 Discussion

6.1 Tools

In this section I would like to point out not only the challenges and learning experience encountered during the project but also the interpretation for the results in Figure 10.

To begin I would like to mention my satisfaction with Kafka. It was very easy to create topics and create consumers and producers to listen and process these streams. I am very excited to see the ability to make high frequency trading simulations with this type of stream processing system. The use of Cassandra became a challenge after realizing how useful SQL is. I had never worked with SQL but after needing to manipulate data from tables in my database and not being able to use SQL for server-side manipulation I had to resort to using Python. Although Python is not hard to use I definitely would have liked the luxury of using SQL for table manipulation.

Tensorflow was very useful and I am very satisfied with it because of how easy the library was to use. Building models and training models on the dataset that I had was of no problem. So many people depend on this library for business and everyday work that there are plenty of issue resolutions and community support for any challenge. I will definitely use this library again for the future work on this project. Plotly with dash was a great tool to use because of the ability to make great graphs in Python so I can check my data and then seamlessly integrate that into a front-end dashboard without any extra work. I hope to master this tool and create a better front-end to show the different strategies

and allocations of not only the current implemented methods but the ones for the future.

6.2 Results Interpretations

I would like to interpret the results that were found during this project. The reason why the GRU models follow the statistical models sometimes for the better and sometimes for the worse is due to the softmax activation. Although the model is not have to be very accurate in predicting the price for a stock in the future time step it still does well because softmax will not consider the accuracy of the prediction but the trend of the prediction. If it was high or low softmax would have caught than and gave a higher portion to that prediction and a lower portion to the others.

7 Future Work

The future work is listed below. I had great fun working on this project and have several directions to explore in the future for this project.

1. Implement Reinforcement Learning Agents: Reinforcement learning agents have been shown to perform very well in complicated tasks to the point of being better than most if not all humans. The idea of reinforcement learning is to maximize the reward given the actions that it can take in an environment off of observations. It this setup the environment is the stock market and the agent can either allocate more towards a stock or less. The observations are the variables in the dataset of stock prices.

2. Transaction Costs: In order to be able to use this system in the real world, transaction costs must be considered. This has been done by other researchers and most commonly only includes subtracting a term that represents the transaction costs on each portfolio adjustment that is not zero.

3. Feature Engineering: The data can be feature engineered to contain the volume of a stock in order for the models to recognize that the stock is being sold heavily and thus our investment will lose value or bought heavily in which our investment will gain value. This can be valuable information to consider in training.

4. Log Returns: Applying a logarithmic function to the returns of a dataset of assets would be beneficial since it then shows a stationary time series usually with mean zero along with having other nice properties for time series analysis methods for both modelling and prediction.

5. Time Series Forecasting: There exist several types of forecasting time series, even specifically for financial data. I would be interested to implement these types of forecasting methods along with the reinforcement learning to see how well we perform.

6. Implement Different Allocation Strategies: Different Allocation strategies might be more beneficial in different types of markets. In an upward trending market a higher risk portfolio may have a higher probability of gaining value without losing money too often. In a downward trending market having a minimum risk portfolio would be most beneficial to reduce the amount of loss during that time.

8 Conclusion

In conclusion, this project aimed to create a dashboard that allows for a user to be able to see different types of portfolio management strategies. Alpha vantage was used to gather data and an Apache Kafka cluster was implemented to process the stream. A Cassandra database was used to store the data for use in modelling later. Two different types of portfolio strategies were considered. One for minimum risk for a target return and the other for most return while taking on minimum risk. Afterward two GRU networks were implemented for each portfolio. A top down investment strategy was used for initial asset allocation and afterward it was left to the statistical methods and machine learning models to manage them. The machine learning models performed similarly to the statistical methods in times of an unstable economy.

9 Resources Used

References

[1] I. Aldridge. Big data in portfolio allocation: A new approach to successful portfolio optimization. *The Journal of Financial Data Science*, 1(1):45–63, 2019.

[2] S. Almahdi and S. Y. Yang. An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems with Applications*, 87:267 – 279, 2017.

[3] A. Cockcroft and D. Sheahan. Benchmarking cassandra scalability on aws — over a million writes per second, Nov 2011.

[4] A. Filos. Reinforcement learning for portfolio management, 2019.

[5] J. Frankenfield. What is a robo-advisor?, Feb 2020.

[6] J. Kreps. Benchmarking apache kafka: 2 million writes per second (on three cheap machines), Apr 2014.

[7] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[8] P. G. MOORE. Mathematical models in portfolio selection. *Journal of the Institute of Actuaries (1886-1994)*, 98(2):103–148, 1972.

[9] I. Staff. What is the ideal number of stocks to have in a portfolio?, May 2019.