Literature Review: Building a Fund for Cryptocurrencies with Machine Learning

Odelia Putterman

March 7, 2022

1 Problem Context

Investing is a means by which people can grow their money exponentially, but it also holds the potential for significant loss, especially in the context of day trading.

1.1 Index Funds: Explained

As it stands, it's difficult, if not impossible, to predict the exact performance of individual stocks. As such, people invest in index funds, collections of stocks which follow some set of rules for construction, to mirror the whole market rather than a particular holding. Such funds are generally mutually pooled, with the stock holdings shared across many holders.

Money that goes into the fund is invested in all the stocks the fund holds, growing the diversification of the fund and minimizing its dependence on individual stock performance. Further, index funds are periodically re-balanced to offset losses and risk, and companies are sold once they leave these predefined rules or parameters discussed.

1.2 Cryptocurrencies: A Unique Opportunity and Approach

Index funds are generally constructed using fundamental measures of a company (aka fundamentals), such as cash flow, sales, and dividends. Cryptocurrencies, however, have none of these intrinsic metrics. Since the cryptocurrencies themselves are a currency rather than a materialized product, there are no sales or dividends to be found. Their only measure is price. Accordingly, any approach to creating and maintaining a cryptocurrency index fund must be revolutionary in nature.

Cryptocurrencies are incredibly volatile, making re-balancing especially important. Their instability also offers a unique opportunity for leveraging their turning points to optimize gains if high-accuracy price prediction can be achieved. If we could predict cryptocurrency prices in advance, we could leverage this knowledge to maximize gains while keeping a sense of security with the breadth of the fund.

Typically, index funds are rebalanced on a consistent, preset schedule, such as the third Friday at the end of each calendar quarterly. But, what if we could predict cryptocurrency prices to decide how and when to rebalance such a cryptocurrency fund instead? If this succeeded, the results would be revolutionary in nature and a first-of-its-kind for maintaining a balanced, high-return cryptocurrency fund.

People have previously attempted to predict the behavior of cryptocurrencies with some success. I aim to follow in their steps with a slightly different approach and following implementations. Such prior work is discussed below under Prior Work.

1.3 Ethical Considerations

This field is endlessly exciting, but there are many ethical considerations that should be discussed and not overlooked.

1.3.1 Obligation to Investors

As an index fund manager, you hold a moral obligation to your holders to (at least try to) grow their investment. Cryptocurrency price prediction carries inherit risk. In this proposed cryptocurrency fund, if you wrongly predict the performance of a cryptocurrency stock, you risk misallocating cryptocurrency stocks and, in turn, losing investor money. Since this fund is reliant on accurate cryptocurrency price prediction which will be, at best, relatively accurate, investors should be forewarned about the risks of investing.

1.3.2 Investing

First off, we must consider the ethical implications of investing itself. It's famously said that "the rich get richer and the poor get poorer". Albeit peoples' ability to turn a little into a lot, it's infinitely harder to become wealthy if you're starting from little and boundlessly easier to become wealthier if you're already wealthy. This leads us to ask, *is investing fair?*

The stock market is not a fair playing field. Because of its inherent compound nature, it favors those who start with more and exponentially returns the more you invest. Moreover, those elite investors have access to tools not readily available to the average person, such as personal wealth managers and access to more exclusive, "high buy-in" funds.

The reality is, we live in a capitalist society. Though it isn't fair that some people start off with more than others, it is a part of the system we subscribe to. To offset the base inequity intrinsic to investing, such a fund as this one discussed could offer low management fees or possibly free management and choose its stock holdings in such a way that the index fund price is affordable, or otherwise allows for fractional shares.

1.3.3 Price Prediction and Access

If this prediction algorithm is, in fact, highly accurate, price prediction poses follow-up questions:

- Is it fair that only some have access to cryptocurrency market predictions?
- Is it right to make cryptocurrency predictions widely available?
- If cryptocurrency predictions are made widely available, will they alter cryptocurrency prices themselves, leading to "false" or "announcement-driven" outcomes?

There are many questions to be asked and contemplated beyond these. They lead me to believe that cryptocurrency price prediction can hold incredible harms, especially if kept exclusive with high subscription costs or made public to the point where the cryptocurrency predictions themselves alter the cryptocurrency prices, becoming a self-fulfilling cycle.

I believe there is a use, if only in the context of research, to learn about the drivers of cryptocurrency prices through sentiment analysis and see their implications for investment holdings, even if the outcome decides solely that these tools are too powerful and should be kept unavailable to the public.

That said, cryptocurrency price predictions aren't any more unethical than current practices. As it stands, the wealthier do have access to tools which help them invest smarter. This tool may not be ethical, but that isn't anything new to the world of investing.

2 Technical Background

I propose to predict cryptocurrency prices using machine learning (ML) and sentiment analysis. Because, as previously mentioned, cryptocurrencies don't have fundamentals, we must evaluate and try to predict their behavior using other metrics for input data. Here, I propose the use of sentiment analysis and current events data.

2.1 Sentiment Data

Sentiment data is essentially the same as it sounds: it is data which signifies sentiment. Such sentiment, in this case, should convey feelings towards or about cryptocurrencies. This can take the form of news articles, social media posts, or other messaging. Such sentiment data will include, but is not limited to:

- The number of positive and/or negative news articles about specific cryptocurrencies/cryptocurrency (as a concept) generally
- The number of positive and/or negative social media posts about specific cryptocurrencies/cryptocurrency (as a concept) generally
- The number of positive and/or negative crypto blog postings about specific cryptocurrencies/cryptocurrency (as a concept) generally

I plan to feed how many of each such posting there is in a given time period (to be determined) as the input data for my machine learning algorithm, with separate datapoints for post type, positive or negative, and specific to a given cryptocurrency or cryptocurrency generally.

This list is not fully fleshed out yet. As I continue my project research, I plan to add more input datapoints to this suggested sentiment data input vector.

2.2 Current Events Data

As we have seen recently with events in the Ukraine, current events have a strong impact on cryptocurrency prices. Accordingly, leaving them absent from cryptocurrency price predictions would be ill-advised. To train and maintain a machine learning model, however, we need to somehow quantify this data. The challenge here is *how.*

I propose, similarly, that we use sentiment to gauge world-events issues. I could add to this input vector of data datapoints which signal stress levels and financial concern with shifting scales, such as 1-5, where a one signifies low stress and a five signifies high stress.

2.3 Obtaining Data

Now that we have determined which data to analyze, we must consider how this data will be acquired. Ideally, there would be preset datasets with this information. But, I have yet to find any with the breadth, depth, and upkeep necessary for such a machine learning algorithm to succeed. So, scraping is my preferred method for training and maintenance data acquisition. I don't want my project to be about how to scrape data, rather I'd like to make this as painless as possible. To prevent scraping itself becoming a massive project, I plan to use existing libraries for help tackling this problem.

Potential Libraries (Python):

- PyGoogleNews
- NewsCatcher
- FeedParser
- NewsPaper3k

Cryptocurrencies are a global phenomenon, so we should not limit our scope of search to US-based sites. But, this is still a college project with limitations to its scope, so I plan to start with sourcing articles written in English from mostly US-based cites, but eventually (if time permits) broaden my scraping to more non-US websites and, potentially, other languages.

I plan to narrow my list of sentiment data sources and scrape from these to get the training data and periodically retrieve classification data for price prediction. I suggest the following sources be tracked:

2.3.1 News Sources

1. The New York Times
2. The Economist
3. CNN

2.3.2 Social Media

1. Instagram
2. Facebook
3. Twitter

2.3.3 Crypto Blogs

1. Cointelegraph
2. NewsBTC
3. CryptoNinjas

Most of these should be available to be openly viewed or have APIs which could be connected for scraping. This scraping would look for occurrences of each of the aforementioned datapoints and add them to an input vector, potentially spanning the course of a few hours to a week (time range to be decided).

2.4 Deciphering Sentiment

To decipher sentiment, I will use an existing natural language processing (NLP) algorithm.

Possible Algorithms (Python):

1. NLTK (Natural Language Toolkit)
2. SpaCy
3. TextBlob
4. Stanford CoreNLP
5. Gensim

2.5 Sequential Support Vector Machine

A support vector machines (SVM) is a type of machine learning algorithm which allows for linear classification of input data. SVMs are supervised learning models which take a training set with output data marked as one of two categories. Support vector machines look for a 1-dimensional line or hyper-plane that separates the two classes, with the "best" dividing line being that which separates them right in the middle, known as the **decision boundary**. As such, SVMs maximize the distance between the two outcomes so as to classify new input data into one of these two groups.

2.5.1 Derivation

In the 1-dimensional case, a dividing line would take the form:

$$y = ax + b$$

where a is the slope, b is the y-intercept, and x is the input data.

If we replace x with x_1 and y with x_2 and define $\mathbf{x} = (x_1, x_2)$ and $\mathbf{w} = (a, -1)$, this equation simplifies to the hyperplane equation:

$$w \cdot x + b = 0$$

With this hyperplane equation, we derive predictions using points above and below the hyperplane. This holds for our multidimensional situation, where w is our vector for our input data. Points above the hyperplane are given the value +1 and points below the hyperplane are given the value -1:

$$h(x_i) = \begin{cases} +1, & \text{if } w \cdot x + b \geq 0 \\ -1, & \text{if } w \cdot x + b < 0 \end{cases}$$

To find the "best" hyperplane, we must find the hyperplane with the greatest margin, as previously mentioned. First, we define $\mathbf{f} = y(w \cdot x + b)$, so that the sign of f is positive if the data is correctly classified and negative otherwise, and divide this by the length of our w vector, to get:

$$\gamma = y\left(\frac{w}{||w||} \cdot x + \frac{b}{||w||}\right).$$

For each datapoint in our training dataset, we calculate this γ value, and minimize it to find M, the *geometric margin*:

$$M = \min\, y_i\left(\frac{w}{||w||} \cdot x + \frac{b}{||w||}\right) \text{ for } i = 1...m.$$

Our "best" hyperplane will be that were M is maximized. To get the w and b values which maximize M, we must solve the optimization problem:

$$\max M = \max \frac{f}{||w||} = \max \frac{1}{||w||} = \min ||w|| = \min \frac{1}{2}||w||^2 \text{ for } f_i \geq 1, i = 1...m.$$

We have turned our maximization problem into a minimization one, which is easier to solve.

We can now use Lagrangian multipliers to solve for w and b. We will not finish this derivation here, but you can learn more at Understanding the mathematics behind Support Vector Machines, from which this derivation was based off of.

2.5.2 Sequential SVMs

As I have demonstrated, SVMs allow for the classification of data into two groups. However, my project aims to predict the price of cryptocurrencies, which is continuous rather than discretely divided. So, I propose using sequential support vector machines to continuously group data into one of two cases, allowing for many more outcomes. Essentially, we would set up a series of support vector machines which, at each step, classify the input data into one of two groups. First, the data would be classified into two groups: rising cryptocurrency price and falling cryptocurrency price. Next, those classified as rising or falling would be split separately into "rising a lot" versus "rising a little" or "falling a lot" versus "falling a little", and so on and so forth. This would need to be quantified somehow, so that "a lot" or "a little" is represented by some percentage, such as a 1% increase or decrease in price. We could add in an error estimate as well for further safety in cryptocurrency price predictions and their weight on the cryptocurrency index fund trading algorithm decisions. In this way, we could get more detailed insight on the predicted price of a cryptocurrency.

Since we wish to predict the price of each cryptocurrency individually to rebalance our index fund, this sequential SVM would be run individually on each coin. The predictions from these sequential SVMs would then be passed to our trading algorithm (described later under Rebalancing Trading Algorithm).

A few questions remain: *Where do we draw the lines between sequential SVMs? Should we choose a percentage of increase or decrease?* I don't know the answers to these questions, but I plan to research more and play around with the results as I further pursue this project.

2.5.3 Useful Libraries

I am currently taking Machine Learning, where we are self-implementing these algorithms, so I am strongly considering coding these sequential SVMs myself. However, this might prove more difficult than I am imagining, so I am also prepared to use already existing libraries for the bulk of this work.

Potential Libraries (Python):

- Scikit-learn
- LIBSVM
- ThunderSVM

2.6 Support Vector Clustering Algorithm

My second thought is to use a support vector clustering algorithm to look for natural clustering in the data rather than assume sentiment analysis is the key deciding factor. A support vector clustering algorithm model is an example of unsupervised machine learning. Rather than be told what causes surges or declines in cryptocurrency prices, it will look for such natural clustering in unlabeled data to identify and predict future cryptocurrency prices.

This second method would serve as an interesting point of comparison to the sequential SVMs to note if our suggested sentiment analysis is viable or a poor predictor for cryptocurrency prices.

Potential Libraries (Python):
The same libraries suggested for the sequential SVMs approach can be used for the support vector clustering algorithm approach.

2.7 LSTM Neural Networks

The other thought I have, which could replace or compliment the use of support vector machines, is using Long Short-Term Memory (LSTM) networks. These networks learn order dependence for prediction problems, which is especially useful for something like stocks where order and history are relevant.

LSTMs are an example of a working recurrent neural network. In some situations we need a lot of context, while in other cases we need only to know what previously happened. LSTMs solve this problem by deciding when to keep and/or use what information. Specifically, LSTMs were developed as a subtype of recurrent neural networks (RNNs) to avoid the problem of depending on outdated inputs.

At each step in the network, a message is passed to the successor step. Rather than throw away previous training input, LSTMs, or rather recurrent neural networks, use previous events to inform coming ones with loops. But, rather than hold onto its training data forever, these recurrent neural networks hold onto input information dependent on the input data weight. It is more fluid than traditional neural networks in its ability to contextually choose relevant information.

To learn more about why LSTMs are useful, how they work, and the mathematics behind them, visit this blog post: Understanding LSTM Networks.

Potential Libraries (Python):

- TensorFlow
- PyTorch
- NeuroLab
- Scikit-Neural Network

2.8 Implementation

2.8.1 Choosing an NLP Algorithm

First off, I believe it is important to use a few NLP algorithms on scraped data to see and evaluate their results. Based on evaluation of these, I will decide the best NLP algorithm to use for finding sentiment data. If there are a few contenders, I propose building a cryptocurrency price prediction model for each one and evaluating their outputted results.

2.8.2 ML Algorithms

I don't know which of the three proposed machine learning algorithms will perform best for cryptocurerncy price prediction, so I am planning to attempt all three and compare their results, potentially building a few models for each with the chosen NLP algorithm(s).

2.9 Rebalancing Trading Algorithm

There are many ways to rebalance an index fund. The usual choice is **calendar rebalancing**. In calendar rebalancing, the index fund holdings are reviewed on a preset calendar basis and adjusted to their original form. The actual rebalancing is done by examining issues such as transaction costs and drifts.

However, my choice of rebalancing would be a mixture of calendar rebalancing and **percentage-of-portfolio rebalancing**. Basically, every given period of time deciphered by my ML algorithms as ideal (i.e. when prices are dramatically changing), I would rebalance the portfolio to mitigate risk but maximize profits with percentage-of-portfolio rebalancing.

2.10 Evaluation

To evaluate the success of these models, I plan to feed the models data from after the training period and compare the outputted cryptocurrency price prediction results against the actual cryptocurrency prices from that time to see if the ML models succeeded.

To evaluate the success of the cryptocurrency index fund itself, I will compare the year-to-date (YTD) success of the machine learning index fund(s) - potentially multiple funds based on different NLP algorithms, ML algorithms, and training datasets - with the YTD success of common cryptocurrency index funds (i.e. ProShares Bitcoin Strategy ETF, Grayscale Bitcoin Trust, Bitwise 10 Crypto Index Fund...).

Based on how each model performed, we can answer some follow up questions to maximize the accuracy of the cryptocurrency price prediction algorithms and index fund:

- How did each perform?
- Should we use on or another cryptocurrency price prediction solely to inform the cryptocurrency index fund trading algorithm?
- Should we average out the cryptocurrency price predictions from the three machine learning approaches to build the cryptocurrency index fund?

2.11 Coding Details

As previously hinted, for this project I will code in Python and use PyCharm as my integrated development environment (IDE). Python is my language of choice for its abundant scraping and machine learning libraries/tools in addition to its simplicity of use.

3 Prior Work

3.1 Forecasting and trading cryptocurrencies with machine learning under changing market conditions

This article, Forecasting and trading cryptocurrencies with machine learning under changing market conditions, closely aligns with the aims of my project. Rather than apply these predictions to a fund with great range in cryptocurrencies, this research paper instead focuses on examining the predictability of three major contenders: bitcoin, ethereum, and litecoin. Moreover, they focus on times of extreme upheaval and use linear models. They used three ML models - linear models, random forests, and support vector machines - for their predictive work. Of the 18 models they built, 13 models had a success rate over 50%, affirming the promise of ML algorithms for cryptocurrency stock price prediction. In their introduction, they explain the proposal of sentiment analysis for stock price prediction, which is what motivated me to choose the same for my project.

3.2 LSTM-based Sentiment Analysis for Stock Price Forecast

A favorite research paper I found was LSTM-based sentiment analysis for stock price forecast. Similar to what I propose to do, this paper uses a long short-term memory neural network with text sentiments and historical stock transactions to predict stock prices. This paper argues in its introduction the importance of aggressive investing strategies and, hence, the usefulness of accurate stock trend forecasting. They use fundamental analysis and technical analysis to forecast stock prices in addition to an LSTM, which was applied to this forecasting.

Using text sentiment from blog articles, social media, online forum posts, and product reviews, the authors aimed to understand online public opinions. They used an existing NLP algorithm (BERT) to classify the texts into sentiment categories by counting the emotional words that appear in a given sentence, then calculating a score for each emotional word, and, with these individual scores, output the condition of the sentence. Further, this article addressing preprocessing of data to get it into a form where it can be passed to the ML algorithm, which is very useful for me to apply to my own product algorithm.

To evaluate their results, the authors created four different forecast models based on different training data and evaluated them using root mean square error (RMSE). From their work, a clear improvement was drawn, up 12.05%.

3.3 Algorithmic Trading

3.3.1 Mean-Variance Optimization

In the Medium article Algorithmic trading based on mean-variance optimization in Python, Markowitz's portfolio optimization (MPT) and the Modern Portfolio Theory are implemented in python as trading strategies with the zipline library. The general goal of MPT is maximize returns while minimizing risk. That said, this method ignores transaction costs, which, for an index fund, do need to be considered.

In this trading algorithm, short-selling is not allowed. However, with my suggested index fund, we might want to allow short-selling. The work presented in this article does allow for rebalancing on a given schedule, which I could utilize to set a schedule based on the ML outputs.

3.3.2 Constant Rebalanced Portfolios

A more likely approach I will take is implementing a *constant rebalanced portfolio*, achieved in this blog post: Constant Rebalanced Portfolios - some simulations with numpy. The idea explained in this article is:

Say we have three stocks, and we believe each will perform equally well, so we split our money in three, making our holdings $\{1/3, 1/3, 1/3\}$. Let's say one of these three stocks does significantly better than the rest, making our holdings: $\{1/2, 1/4, 1/4\}$. The idea here is we believe this disproportionate return was just chance, and so we sell off some of this stock and rebalance to our original $\{1/3, 1/3, 1/3\}$.

With my index fund, what we might say is:

Stock A is planned to increase in price, so let's buy a lot of it before it has increased. Once it has increased, let's sell it off and regain our original balance.