Optimal Cryptocurrency Portfolio Selection Using Genetic Algorithm

Gursahib Singh

Electrical and Computer Engineering, University of Waterloo

Student Id: 20942046

Abstract— **The new age world is now getting oriented towards cryptocurrency. But there are certain unexplored risk factors attached to it which can be very harmful for the investment. In this project, Genetic Algorithms are used to optimize a cryptocurrency portfolio. The optimizations are carried out using various parameters. The methodology to form the initial population and fitness function is described in detail. The Genetic Algorithm techniques such as crossover and mutation are discussed in the report. Certain experiments which are carried out for different parameters are described and results are critically discussed along with suitable graphs and figures to know as to what factors contribute the most while deciding the portfolio. There is a detailed discussion on the results and also the future work about what can be done in order to extend the project further. The future work discusses techniques to further increase the efficiency. Appendix contains the code written in python for demonstration.**

Keywords— **Portfolio, Covariance, Crossover, Mutation**

I. INTRODUCTION

Genetic algorithms are unique ways to solve complex problems by harnessing the power of nature. Unlike artificial neural networks (ANNs), designed to function like neurons in the brain, these algorithms utilize the concepts of natural selection to determine the best solution for a problem [1].

Genetic algorithms generally follow a brute force approach to reach to a solution. Brute force solutions are time consuming and to speed up the process, they use the concept of evolution. By applying these methods to predicting portfolios, traders and investors can optimize trading rules by identifying the best values to use for each parameter for a given security. In the financial markets, genetic algorithms are most used to find the best combination values of parameters in a trading rule [1].

II. LITERATURE REVIEW

The possibility of utilising Genetic Algorithms to forecast the momentum of stock price has been previously explored by many optimisation models that have subsequently addressed much of the scepticism. But not much research has been put to predict the cryptocurrency market. Work is in place to investigate the dependencies of cryptocurrency market. The different varieties of factors include asset type and pricing, hedging, and market efficiency [2].

Portfolio optimization is the process of selecting the best portfolio (asset distribution), out of the set of all portfolios being considered, according to some objective. The objective typically maximizes factors such as expected return and minimizes costs like financial risk. Modern Portfolio Theory was introduced in a 1952 doctoral thesis by Harry Markowitz. It assumes that an investor wants to maximize a portfolio's expected return contingent on any given amount of risk. For portfolios that meet this criterion, known as efficient portfolios, achieving a higher expected return requires taking on more risk, so investors are faced with a trade-off between risk and expected return [3].

Several researchers have focused on technical analysis and using advanced math and science. Some models have been proposed and implemented using the above mentioned techniques, the authors of Tsang, P.M., Kwok,P., Choy, S.O., Kwan, R., Ng, S.C., Mak, J., Tsang, J., Koong, K., and Wong, T. made an empirical study on building a stock buying/selling alert system using back propagation neural networks (BPNN), their NN was codenamed NN5. The system was trained and tested with past price data from Hong Kong and Shanghai Banking Corporation Holdings over the period from January 2004 to December 2005. The empirical results showed that the implemented system was able to predict short-term price movement directions with accuracy about 74% [4].

III. MOTIVATION

It has been more than 10 years since the emergence of first cryptocurrency-Bitcoin, but still, the world is a bit tentative about the cryptocurrency market. Although, the market promises high returns, but a greater risk is involved. The youth is attracted towards this asset and therefore, there's a need to optimize the returns by correctly tracking a cryptocurrency portfolio.

The primary difference between a stock market and cryptocurrency market is that stock market shuts down for several hours and cryptocurrency market doesn't. Due to this nature, cryptocurrency prediction seems to be more consistent and the same principles and algorithms that are used for prediction of stock prices may not suite for cryptocurrency market.

Thus, there's significant scope and motivation to explore the cryptocurrency market using an artificial life technique such as Genetic Algorithm.

IV. HYPOTHESIS

The optimized portfolio depends upon variety of factors such as no. of assets in portfolio, no. of iterations performed and size of initial population.

V. TOOLS AND LIBRARIES

The primary programming language used is Python. The code is written, run, and tested using Jupyter Notebook. Jupyter Notebook provides interface to test sections of code individually which makes the debugging process easier. Other libraries used are Numpy, Pandas and Matplotlib. Numpy is the library for n-dimensional arrays. Pandas provides useful methods to manipulate the data stored in 2-D *DataFrames*. Matplotlib is the charting library to visualize the data.

VI. METHODOLOGY

With a fitness function in hand, work can then be done to optimize the fitness function by using the natural principles of Genetic Algorithm: Natural Selection, Crossover and Mutation. Other data filtering techniques will also be used on raw data for consistency.

The current project will focus on limited number of assets existing in a portfolio. Each asset has a parameter *weight* attached to it which denotes the amount of allocation of that asset in a portfolio. The *weight* of the asset will take any value ∈ [0,1].

The important parameters are expected return of portfolio and expected variance. These parameters will form the basis of the fitness function which will be optimized further.

The program runs for and compares portfolios with different number of assets ranging from 2 to 6.

A. Data Collection

For this project, the requirement was to calculate expected returns of the cryptocurrency asset. The expected returns of each asset could then be used to predict the expected return of the entire portfolio. The data of prices for cryptocurrency is taken from a public dataset from *Kaggle*. Each *.csv* files contains historical prices for a different cryptocurrency. These historical prices are used to calculate the expected returns and variance. Building a Machine Learning model to calculate future prices of assets is beyond the scope of this project

The price of an asset has a unit attached to it. Therefore, it is not an appropriate measure as one cryptocurrency can be represented in term of different units. Instead of the prices, the percentage change in the prices is considered as percentage is unitless and can be applied to any conversion unit.

B. Fitness Function

For an optimal portfolio, the goal is to maximize the returns(profit) and minimize the variance(risk). This would form the basis of an ideal investment strategy.

The expected return of the portfolio is the summation of product of expected return of each asset with its weight in the portfolio as shown in (1).

$$E(p) = \sum_{i=1}^{n} E(a_i) w_i \quad (1)$$

where $E(a_i)$ is the expected return of asset i and w_i is the weight of the asset in the portfolio.

It is also important to calculate the volatility of each asset and the entire portfolio. The volatility of each asset is directly linked with the risk it possesses. Minimization of risk is also important along with maximizing the profit.

Volatility is nothing but the standard deviation of the stock. It is defined as:

$$sd = \sqrt{\sum_{i=1}^{N} (a_i - \bar{a})^2 \Big/ N - 1} \quad (2)$$

where N is the number of samples a_i is the i^{th} value of sample and $\bar{a}$ denotes the mean of the values.

There is another important parameter that needs to be taken into consideration. In most cases, the stock market fluctuates based on a sector. As an example, the finance sector will fluctuate meaning the companies categorized

in the financed sector will often observe the same movement as the sector itself. That means there is some interrelationship between the assets that need to be identified. Because of this interrelationship, fluctuation in one asset could affect the price of another asset.

The parameter to calculate this interrelationship is called Covariance. In probability theory and statistics, covariance is a measure of the joint variability of two random variables. If the greater values of one variable mainly correspond with the greater values of the other variable, and the same holds for the lesser values (that is, the variables tend to show similar behavior), the covariance is positive. In the opposite case, when the greater values of one variable mainly correspond to the lesser values of the other, (that is, the variables tend to show opposite behavior), the covariance is negative [5].

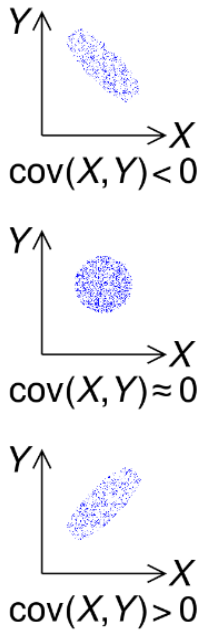

Fig. 1: Plot for variance in different scenarios. Adapted from [5]

The covariance of all assets against each other would be represented in 2-D array with rows and columns represented by assets themselves. The diagonal of the array would represent the covariance of an asset with itself which essentially is the variance. This covariance matrix can be further utilized to calculate the covariance of the entire portfolio. The covariance of the entire portfolio would require multiplying the appropriate weights of the assets as well. Covariance of portfolio is defined as:

$$Cov(p) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j Cov(a_i, a_j) \quad (3)$$

where w_i and w_j are weights of the assets and $Cov(a_i,a_j)$ is the covariance between i^{th} and j^{th} asset. $Cov(a_i,a_j)$ can be taken from the covariance matrix.

The portfolio return and portfolio variance can separately form two fitness functions. The objective of the fitness function that considers expected returns would be maximized. Whereas the fitness function for expected variance would be minimized.

If these two parameters are optimized separately, that would create a problem of compatibility between the weights. For e.g. The set of weights returned by expected return fitness function may not work best for expected variance fitness function and vice-versa. Therefore, there should be a third parameter that could take into account the effect of both returns and variance. This new factor can then solely be optimized for the optimal set of weights and eventually the ideal portfolio.

A parameter that links both returns and variance is Sharpe ratio. The Sharpe ratio was developed by Nobel laureate William F. Sharpe and is used to help investors understand the return of an investment compared to its risk.1 The ratio is the average return earned in excess of the risk-free rate per unit of volatility or total risk. Volatility is a measure of the price fluctuations of an asset or portfolio [6].

Subtracting the risk-free rate from the mean return allows an investor to better isolate the profits associated with risk-taking activities. The risk-free rate of return is the return of an investment with zero risks, meaning it's the return investors could expect for taking no risk. The yield for a U.S. Treasury bond, for example, could be used as the risk-free rate [6].

The formula for Sharpe ration is defined as:

$$Sharpe\ ratio = \frac{R_p - R_f}{\sigma_p} \quad (4)$$

where R_p is the return of the portfolio, R_f is the risk-free rate (constant value) and σ_p is the standard deviation of the portfolio.

The risk-free rate being a constant value won't impact the fitness function and can be skipped from the calculations. Finally, the fitness function represented in terms of (1) and (3) is:

$$Fitness\ Function = \frac{E(p)}{Cov(p)} \quad (5)$$

C. Initial Population

Generating the initial population is the first step in the implementation of Genetic Algorithm. The program

would generate several different portfolios and the best amongst those would be the final and optimal portfolio.

A *Portfolio* is nothing but a Python class which has the following properties:

1) Weights: The weights will be represented as a 1-D array. This is analogous to a chromosome. As chromosome contains the important genetic information, the same way 1-D array would contain the weights for each of the asset present in the portfolio.

W_1	W_2	W_3	…	W_n

Fig. 2: Representation of a portfolio as a chromosome

2) Fitness: This property will contain the fitness value of the corresponding portfolio. It is important to store the fitness value as this will be compared with future generations to preserve only the fittest chromosomes.

After defining the class *Portfolio*, the next step is to generate some portfolios which will constitute the initial population. For this, there could be random number *r* defined for population size. The program will run for *r* number of times to generate a chromosome and the chromosome will be stored in a list *Pop*.

A chromosome can be generated by generating random numbers between 0 and 1 and storing them in a 1-D array. A careful point to note would be that the sum of the weights should be 1.

$$\sum_{i=1}^{n} w_i = 1 \tag{6}$$

D. Natural Selection

For the process of natural selection, the best portfolio amongst the initial population is chosen. But since this is portfolio would just be generated out of random numbers, therefore, the fitness is not guaranteed to be optimal. Although the selected portfolio would have the best fitness but still more optimizations would be required.

E. Crossover

This is one of the most important steps of the program which is the core of the Genetic Algorithm. In terms of biology, crossover occurs when two chromosomes, normally two homologous instances of the same chromosome, break and then reconnect but to the different end piece [7].

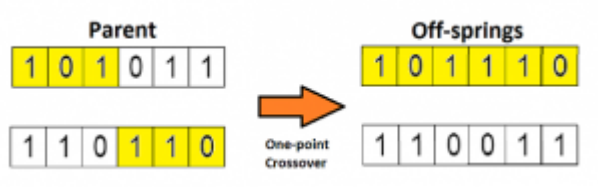

Fig. 3: Representation of crossover. Adapted from [8]

From Fig. 3, it can be observed that a random point is selected. The offspring is generated by combining the genetic material of *parent1* and *parent2* across the randomly selected point.

After the process of natural selection and choosing the best 2 parents. The program will then perform crossover on those parents to generate the offspring. The weights will be adjusted in the offspring in the same way as shown in figure.

After generating the offspring, the fitness value is calculated for the offspring. If the fitness value turns out to be more than the parents, the offspring will replace their parents in the population. The principle which prevails here is “Survival of the fittest”.

There could be a 1-point crossover or n-point crossover. This depends upon the size of the chromosome. If the size of the chromosome is less than 5, it is ideal to go with 1-point crossover.

Another important factor in the crossover step is the crossover rate. Genetic Algorithms follow a brute-force strategy. Therefore, it is not guaranteed that the offspring would be optimal than the parents. If crossover is performed at every iteration, there may be a wastage of computational power as the offspring generated may not be better than parents. A crossover rate with a value between 0 and 1 can be chosen randomly and fixed. During each iteration, before the crossover happens, a random number between 0 and 1 is generated. If the number is less than or equal to crossover rate, the crossover is performed, otherwise skipped.

F. Mutation

After the crossover step is performed, further optimization can be achieved from *Mutation*. In terms of biology, mutations are changes in the genetic sequence, and they are a main cause of diversity among organisms. These changes occur at many different levels, and they can have widely differing consequences. In biological systems that are capable of reproduction, we must first focus on whether they are heritable; specifically, some mutations affect only the individual that carries them, while others affect all of the carrier organism's offspring, and further descendants. For mutations to affect an organism's descendants, they must: 1) occur in cells that

produce the next generation, and 2) affect the hereditary material [9].

One parent from the population is selected at random and a 2-point mutation is performed on that. 2 points (from chromosome) are selected at random and the value of weight at those points are swapped and thus causing a mutation in the weight sequence.

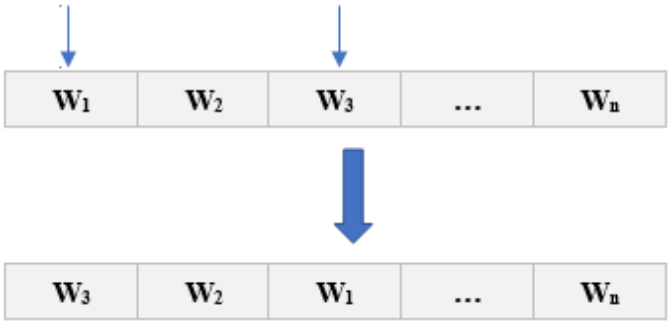

Fig. 4: Representation of 2-point mutation

Just like crossover rate, the program also considers mutation rate as well. A mutation rate with a value between 0 and 1 can be chosen randomly and fixed. During each iteration, before the mutation happens, a random number between 0 and 1 is generated. If the number is less than or equal to mutation rate, the mutation is performed, otherwise skipped. This is done just to avoid saturation of mutations in offspring.

If the offspring has more fitness than the parent, the parent is replaced with the offspring in the population.

VII. EXPERIMENTS

Different types of experiments were performed by tweaking various parameters. The parameters taken into consideration were population size, number of iterations, number of assets.

A. *Comparing Genetic Algorithm techniques*

First, the number of assets is fixed to 6, number of iterations fixed to 100 and population size is fixed to 100. After this the program is run to compare the fitness value as a result of natural selection, crossover and mutation.

B. *Comparing number of assets*

Number of iterations fixed to 100 and population size is fixed to 100. The fitness value is considered after mutation process. Number of assets from 2 to 6 are compared to observe what results they show.

C. *Comparing number of iterations*

Number of assets fixed to 6 and population size is fixed to 100. The fitness value is considered after mutation process. Number of iterations from 0 to 100 are compared to observe what results they show. The results are recorded for every 10^{th} iteration.

D. *Comparing with initial population size*

Number of assets fixed to 6 and number of iterations is fixed to 100. The fitness value is considered after mutation process. Population size from 10 to 100 are compared to observe what results they show. The results are recorded at the increase of every 10 individuals in the population.

VIII. RESULTS

The results are compared for 3 cases namely the case which takes into account only natural selection process i.e., the best chromosome is selected from the randomly generated chromosomes for the initial population. Another case is using crossover with natural selection. The third case is combining mutation with crossover.

Because of the randomness of values and brute-force approach of Genetic Algorithm, it is not guaranteed to have the same fitness value (or same optimal chromosome) on each iteration. Therefore, it is important to observe the trend in the graphs rather than comparing different graphs based on fitness value. The results from these three cases are described below.

A. *Returns v/s Variance*

The returns and variance when visualized by fixing parameters such as population size, number of assets and number of assets show clustering of data points. The point where return is maximum depicts approximately a middle value of variance and the point where variance is maximum shows a point in lower half of the range for returns.

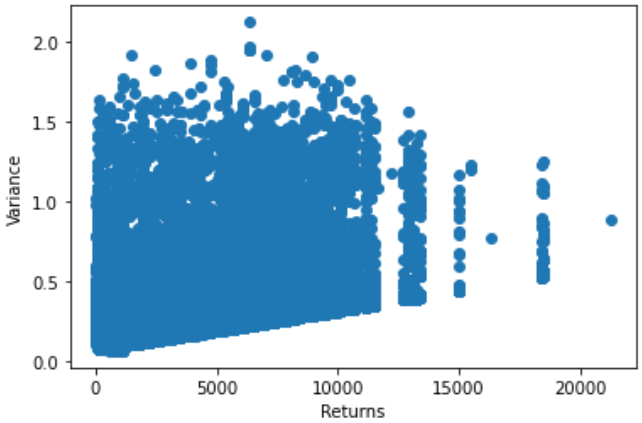

Fig. 5: Graph for portfolio returns v/s portfolio variance

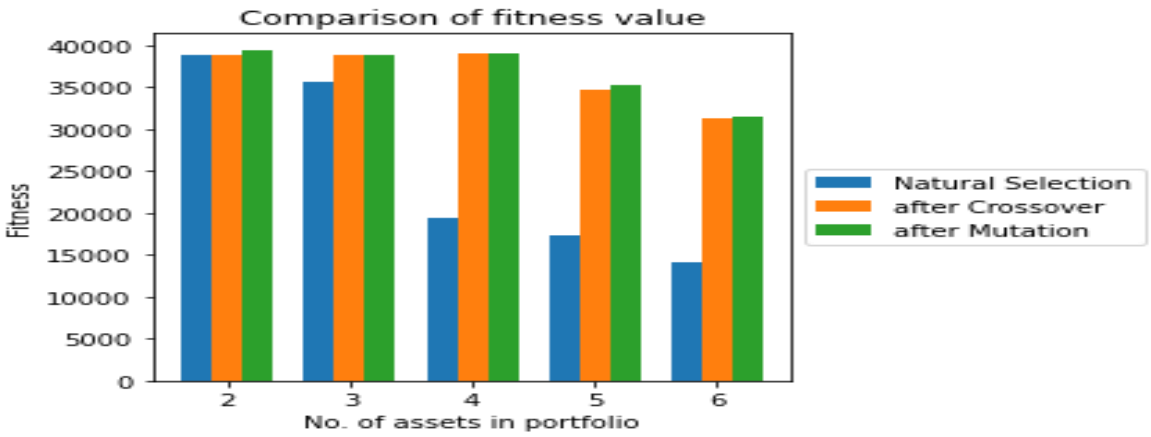

Fig. 6: Graph for comparison of optimization performed by crossover and mutation after natural selection.

B. Natural Selection

The results from natural selection rarely give the optimized result or even close to the optimized result. If the number of assets is less, natural selection most of the times gives the chromosome close to the optimal one but this doesn't happen with a greater number of assets and further optimization is required.

C. Natural Selection with Crossover

There was a significant improvement in the result after using crossover along with natural selection as shown in Fig. 6. The fitness value of best chromosome was 10-20% more than the best chromosome from initial population. Therefore, this gives a clear indication that Genetic Algorithm are working for this case.

D. Natural Selection with Crossover and Mutation

There was slight improvement when mutation was combined with crossover as shown in Fig. 6. Most of the optimization is performed at the crossover step. Mutation just causes a 5-7% increase in the fitness value.

E. Comparison based upon number of assets

This set of comparison is the most crucial comparison. The results show that the lesser the number of assets in the portfolio, the more is the fitness value and hence more optimal portfolio as shown in Fig. 7. Although mutation and crossover do significantly improve the results, but maximum fitness keeps on decreasing with increasing number of assets in portfolio.

$$Fitness \propto \frac{1}{No.of\ assets} \quad (7)$$

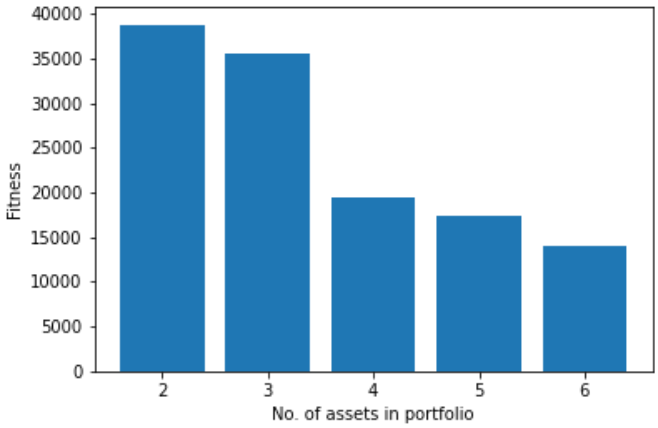

Fig. 7: Graph for comparison of fitness value after natural selection v/s no. of assets

F. Comparison based upon number of iterations

The program compares the fitness value for a range of iterations starting from 10 to 100. The graph is plotted for every 10th iteration as shown in Fig. 8.

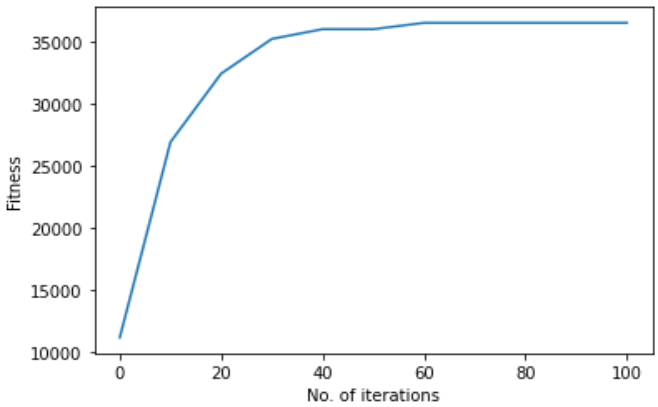

Fig. 8: Graph for comparison of fitness value against number of iterations

G. Effect of initial population size

The results show that more initial population gives better results. The program compares population sizes from 10 to 100 in step of 10. It shows that a better natural selection is made when population size increases as shown in Fig. 9.

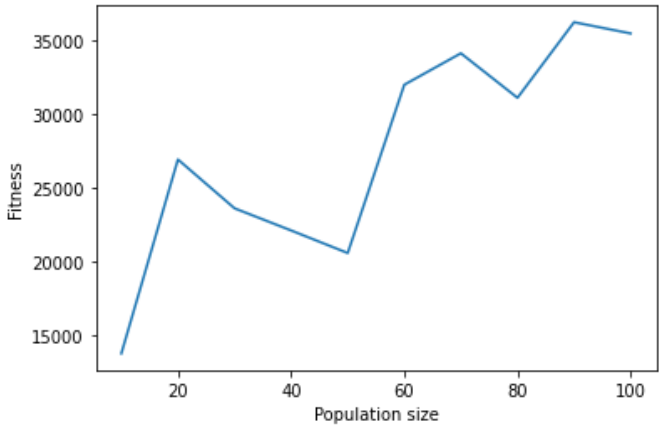

Fig. 9: Graph for comparison of fitness value against number of iterations

IX. DISCUSSION

The results show that Genetic Algorithm techniques such as crossover and mutation produce more optimal results as compared to natural selection. Mutation optimizes just slightly but majority of optimization is done through crossover.

The plot for returns against variance shows the formation of clusters and a linear trend line. The size of the cluster reduces moving along the trend line towards increasing returns and variance. The graph depicts that majority of data lies between the point where returns and variance are minimum and the point where returns and variance approach the middle value of range.

It is a clear indication that as the number of assets increase in the portfolio, the fitness value decreases. The reason for this is the increased variance between the assets and hence the overall portfolio. There is a sharp decline in the fitness value when number of assets change from 3 to 4.

The plot for fitness value against number of iterations indicate that as the number of iterations increase, the fitness value gets optimized. The fitness value raises sharply when the number of iterations ranges from 0 to 60. But after 60 iterations, the fitness value converges. This is an indication of premature convergence.

There are contrasting results for the graph of population size against fitness value. Although the final fitness value is surely way more than the initial one, but the fitness value goes through a lot of corrections before reaching the maximum value. This produces the regions of many crests and troughs. It can be concluded that increase in population size is not always guaranteed to produce better fitness. The reason for this lies in the fact of randomness. It depends upon the fitness of the randomly generated initial population. If the fitness of the initial population is decent enough, increase in population size would affect the fitness positively. But this case can also lead to premature convergence as there won't be many optimizations to make.

X. FUTURE WORK

The project can be extended further to incorporate more features and even improved results. The technique can be extended for *n* number of assets that an investor may want to keep in the portfolio. The value of *n* can be dynamically input at runtime.

This project doesn't assume a 0 weight of asset. Another improvement will be to ignore the asset that increases the portfolio covariance. This would be part of the dimension reduction problem where less significant features are removed from the dataset.

Another problem that's observable with Genetic Algorithms is premature convergence. To solve the problem of premature convergence for consistent results, different techniques can be applied. One of them is increasing population size. Another is to reshuffle the population after certain number of iterations.

The project can also be extended to use Machine Learning models to predict future prices and then get the expected returns of the assets and eventually the portfolio. Because the investment is done considering the future so it would be viable to predict the future prices and then decide the best portfolio.

XI. CONCLUSION

It can be concluded form the experiments and results that hypothesis stated in the report is satisfied. Genetic Algorithm surely help to optimize a given cryptocurrency portfolio. Using the above-mentioned methodology, an investor can estimate the amount of investment to make in a certain asset. The weights of the asset will be optimized in order to result in the best portfolios.

However, careful considerations should be made regarding the parameters used during the prediction. These parameters are population size, number of iterations, number of assets. All these parameters affect the optimization of the portfolio.

It can also be concluded that crossover improves the results significantly while mutation only does a slight increase in the optimization.

ACKNOWLEDGMENT

This project would not have been possible without the guidance of Prof. Chrystopher L. Nehaniv and Prof. Kerstin Dautenhahn. I would also like to extend my gratitude to my parents who have been a source of strength.

REFERENCES

[1] J. Kuepper, "Using Genetic Algorithms to Forecast Financial Markets.", May 13, 2020 [Online] Available:https://www.investopedia.com/articles/financial-theory/11/using-geneticalgorithms-forecast-financial-markets.asp

[2] R. P. Kanungo, "Genetic Algorithms: Genesis of Stock Evaluation.", March 2004 [Online] Available:https://papers.ssrn.com/sol3/papers.cfm?abstract_id=535242

[3] "Portfolio optimization", Wikipedia.org, June 7,2021 [Online] Available: https://en.wikipedia.org/wiki/Portfolio_optimization

[4] D. Purkayastha, "Stock Market Prediction Literature Review and Analysis", Techno College of Engineering Tripura University, June 2019 [Online] Available:https://www.academia.edu/39444154/STOCK_MARKET_PREDICTION_LITERATURE_REVIEW_AND_ANALYSIS_A_PROJECT_PROGRESS_REPORT_Submitted_by_DIPANKAR_PURKAYASTHA_Under_the_supervision_of

[5] "Covariance", Wikipedia.org, October 25,2021 [Online] Available: https://en.wikipedia.org/wiki/Covariance

[6] J. Fernando, "Sharpe Ratio", Investopedia.com, October 4 2021 [Online] Available: https://www.investopedia.com/terms/s/sharperatio.asp

[7] "Chromosomal crossover", sciencedaily.com [Online] Available:https://www.sciencedaily.com/terms/chromosomal_crossover.htm

[8] S. Jain, "Introduction to Genetic Algorithm & their application in data science", Analytics Vidhya, 31 July, 2017 [Online] Available:https://www.analyticsvidhya.com/blog/2017/07/introduction-to-genetic algorithm

[9] Loewe, L., "Genetic mutation", Nature Education, 2008 [Online] Available:https://www.nature.com/scitable/topicpage/genetic-mutation-1127/

APPENDIX A

A Python file for the demonstration of the project is attached with the report. The program using libraries such as *numpy*, *pandas*, *matplotlib*, *random* and *deepcopy*. All the datasets are taken from *Kaggle* project. The datasets are licensed under **CC0: Public Domain**

Link: https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory/metadata

The data for 6 cryptocurrencies is loaded. The data is considered for the year 2021. There are separate *DataFrames* created for assets ranging from 2 to 6. For each of the *DataFrame*, covariance matrix is computed after taking the percentage change of the values. After computing the covariance matrix, the individual returns for crypto currencies is computed. After this step, annual standard deviation is calculated for each of the *DataFrame*.

With all these parameters in hand, functions for Genetic Algorithm are created. The variables for population size, population, no. of assets, iterations, crossover rate, mutation rate, best chromosome are initialized. A class called *Portfolio* is created which will have 2 data members: chromosome and fitness. The function *Init()* will initialize a population. The function *Crossover()* and *Mutation()* choose parents randomly on each iteration and perform operations of crossover and mutation respectively. If the offspring has more fitness than the parent, it is replaced in the population. There is a function *MemoriseGlobalBest()* to get the best set of chromosome from the population.

For each *DataFrame*(based upon no. of assets), the program runs a loop and computes the value of the best chromosome to get the optimized portfolio. A set number of iterations is performed for each *DataFrame*. After computing the results, several graphs are plotted using the *Matplotlib* library. These graphs are plotted to compare how different parameters affect the fitness value.

APPENDIX B

A short video demonstration of the project is also attached along with report.