Predicting Return on Investment for Financial Data Using Different Machine Learning Modeling Techniques

Martin Boyle, Ben Jochem, Brian Kasper
{mzb400, bmj5307, bck5168}@psu.edu

***Abstract*— QUANTITATIVE methods have long been used to understand and predict price movements in financial markets. Traditional statistical techniques have successfully identified and defined various common relationships within and between financial assets leading to an increased understanding of market structures. However, these traditional techniques are often too simplistic to formulate accurate data representations of market dynamics at different times and in different contexts. Thus, they are limited in their effectiveness when forecasting prices and returns in dynamic markets. Machine learning techniques have successfully been used to overcome this limitation (at the expense of interpretability) and are widely used in the field today. Our project investigates the use of various machine learning techniques for forecasting an investment's rate of return based only on historical market data.**

I. Introduction

The purpose of our project is to investigate the effectiveness of various machine learning techniques and algorithms for predicting an investment's rate of return based only on historical market data. Specifically, our goal is to determine the effects that different modeling techniques have on the performance of our end model. Therefore, we are not attempting to only achieve the highest prediction accuracy on our data set (although we will attempt to do so) but to understand how our different approaches impact the end models. This purpose is intentionally non-specific such that our results can be generalized to other markets and used to better understand machine learning techniques for predicting investment returns in any context. One such emerging market where this research is likely to be useful is in the cryptocurrency markets. Cryptocurrency markets are structured in a fundamentally different way than traditional financial markets. This poses the question of whether the same machine learning modeling techniques that have been successfully used to predict market movements in traditional markets can also be used to predict market movements in cryptocurrency markets or whether new approaches are needed. Given that there has been few prior works investigating this problem, our research focuses on making some headway. We apply various different machine learning modeling techniques to data from both traditional markets and cryptocurrency markets. This allows us to analyze the results and determine whether or not the various techniques exhibit the same predictive tendencies in the two markets. Of course, any two markets (even if they are both traditional markets) differ in their structure and modeling techniques will therefore vary in their effectiveness. However, it has many times been exhibited that different kinds of machine learning modeling techniques perform best with certain kinds of data or in specific contexts. For example, it is widely known that the Long Short-Term Memory recurrent neural network model consistently outperforms other architectures when dealing with time series data in financial markets. It is an open question as to whether this holds for novel cryptocurrency markets. This is only one example of what our investigations may shed light on and we intend for this research to provide those that are interested in predicting rates of returns in any financial market with useful rules of thumb regarding when to utilize certain modeling techniques. Figure 1 below gives an overview our solution framework for determining the effectiveness of various machine learning techniques and algorithms for predicting an investment's rate of return in both traditional markets and in cryptocurrency markets.

Beneficiaries of our project include trading firms, individuals, and researchers that have an interest in machine learning applied to financial data. By understanding the impacts that the type of machine learning model and techniques used have on our predictions for different assets in different market conditions they will be able to better decide which approaches they may want to use for their task. This is especially relevant in esoteric or emerging financial markets where asset price patterns are less established or correlated.

II. Related Works

Deep Learning with Long Short-Term Memory (LSTM) - [4]

This article goes into the application of Long Short-Term Memory (LSTM) onto financial data to assess how well it applies to this subject. It also compares this to other algorithms such as linear regression, and others. Lastly, they look into common practices of stocks and find patterns in the data. We decided to choose this resource because it is closely related to our goals and what we hope to accomplish. We want to create a model specifically for cryptocurrency and research more on that topic specifically, but this is mostly what our topic includes. Our plan is to use this knowledge of LSTM models to create a best model for our data and then eventually be able to apply that model and perfect it to cryptocurrency data.

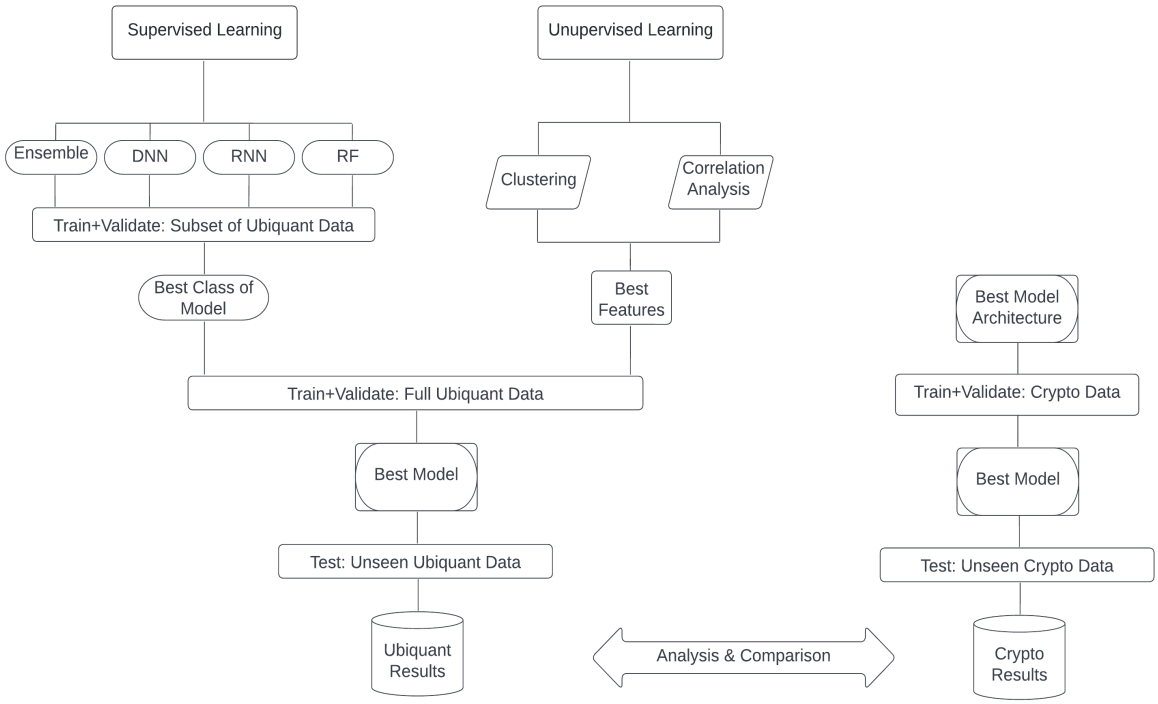

Fig. 1. Solution Framework Pipeline - A high level overview of the solution framework that we use in this study the effectiveness of various machine learning techniques and algorithms for predicting an investment's rate of return in a traditional market and a cryptocurrency market. We first attempt to find the best modeling technique for predicting an investment's rate of return in a traditional market setting using the Ubiquant dataset. The best modeling technique is a combination of supervised and unsupervised learning techniques. A detailed description of the methods used can be found in the solution framework section. After, finding the best model on the Ubiqant dataset and testing it on unseen data to obtain results, we use the same model architecture to train and test on cryptocurrency data. We then compare and analyze the results from the two datasets to determine how the model behaves in the two different contexts.

Clustering Patterns in Cryptocurrency - [6]
This resource analyzes the entire cryptocurrency market by analyzing the daily price time series of 437 cryptocurrencies. They specifically talked about permutation entropy over time windows of price log returns. We decided to choose this because we are looking to acquire cryptocurrency data to run our model on and our model needs to be perfected for this data. We plan to get about 1-2 weeks of data, from every day of those weeks. Our plan for this resource is to use the knowledge on the cryptocurrency market to further our model and further the data collection process for our final model.

An approach to predict and forecast the price of constituents and index of cryptocurrency using machine learning - [2]
This paper investigates the effectiveness of four different machine learning techniques for predicting cryptocurrency. The four different techniques they used were gradient boosted trees, neural nets, ensemble methods, and KNN. Using the root mean squared error, prediction trend accuracy, absolute error, relative error, squared error, correlation, and squared correlation as metrics the author's were able to find models comparable to state of the art LSTM models in prediction efficacy. This paper offers substantial guidance for the cryptocurrency portion of our project. It summarizes the various methods that have been studied for cryptocurrency price prediction and offers a comprehensive comparison to its own methods. This comparison allows us to better understand the tradeoffs between the different techniques in this context and will save us time from implementing inefficacious models. Our plan for this resource is to take the state of the art research that is distilled and expanded upon in this article to formulate our approach for building our model for cryptocurrency price prediction. By improving upon techniques that were deemed promising in this paper we will be able to zero in on only the most efficient models for this task and see how they can be employed in different forms and what this results in.

Prediction of stock price using LSTM - [3]
In this paper, the authors use a LSTM model. However, this paper also explains a deep recurrent neural network. It is essentially the same as LSTM, but there is a dropout method. This dropout method randomly deletes neurons in each layer with a certain probability. This dropout method is used to try and prevent over-fitting. This could be very useful in our LSTM model because overfitting could become a problem with our data.

The paper also describes an associated neural network model. This model has three different branches for the opening price, lowest price, and highest price. Instead of having these branches be predicted by separate networks, the associated neural network model combines all three at different stages. So, the model will predict the opening price and combines it with the LSTM network of the predicted lowest price branch to

help get a better prediction, and then both the first and second branch are used to help predict the third branch for the highest price. This can be incredibly useful for this project for using different predictions of features that have some relevance to each other to help strengthen the prediction of other features. Below is a figure from the paper that visualizes the associated neural network.

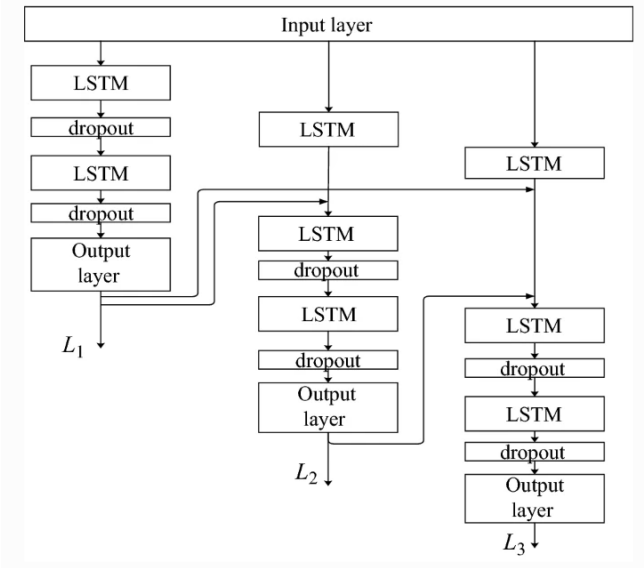

Fig. 2. Associated Neural Network Structural Model

A deep learning framework for financial time series using stacked autoencoders and long-short term memory - [7]

This research article covers the application to deep learning on financial approaches using wavelet transforms, stacked autoencoders, and LSTM. It also covers the three most popular forms of deep learning approaches, which are convolutional neural networks, deep belief networks, and stacked autoencoders. They lastly test the WSAEs-LSTM in multiple financial markets instead of one.

Seen in figure 3, we see the architecture of a LSTM memory cell. This figure is quite useful for us because before this project, we didn't know what an LSTM model consisted of. This helps us understand the memory cell, and how it remembers inputs based on time, and forgets after a certain time. Along with this, it also explains the LSTM repeating module which is a lot of memory cells and other cells meshed together.

III. Methodology

A. Data Collection

For this project we require two sets of data. One to build our models and another to test our models using data with known features. The data for training our model has been collected from Ubiquant Investment through Kaggle.com [1]. To help test our models we will be using data for cryptocurrencies. This data will be collected from Yahoo Finance. We originally intended to collect data daily for about 10 different cryptocurrencies, but we would did not have enough time for

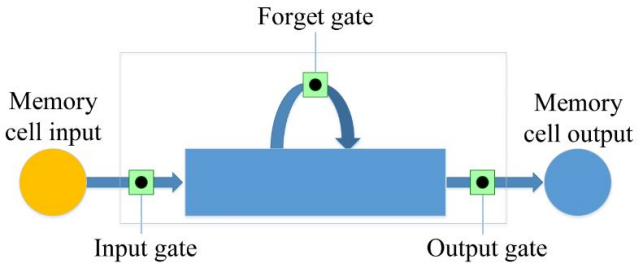

Fig. 3. The Architecture of an LSTM Memory Cell

this collection. Instead, we used historical data from Yahoo Finance for Bitcoin. We chose Bitcoin because it is at the top of the cryptocurrency market. Bitcoin is considered a reliable cryptocurrency which is why we chose it rather than another currency, because there are currencies termed "meme-coins" which have no value besides the joke they provide. A cryptocurrency like Bitcoin has value for the technology it uses for security and with its limited availability. Meme-coins might have unlimited availability and provide very little for security and because of this they can be almost impossible to predict as their value is purely from the value of the meme that they provide. The dataset includes columns for date, open price, close price, high price, low price, closing price, adjusted closing price, and trading volume for 366 consecutive days in 2021.

Date	Open	High	Low	Close	Adj Close	Volume
4/20/2021	55681.79	57062.15	53448.05	56473.03	56473.03	6.78E+10
4/21/2021	56471.13	56757.97	53695.47	53906.09	53906.09	5.49E+10
4/22/2021	53857.11	55410.23	50583.81	51762.27	51762.27	7.48E+10
4/23/2021	51739.81	52120.79	47714.66	51093.65	51093.65	8.67E+10

Fig. 4. An example of the Bitcoin data that is collected

B. Solution Framework

To formulate a solution we need to have a strong understanding of different factors that can help shape financial markets and features for predicting an investment's rate of return. As stated previously, the dataset we use is ambiguous so we need to test for feature importance because our domain knowledge is limited as we do not know what the features are.

We use a variety of both supervised and unsupervised learning models on the Ubiqant dataset. Feature engineering is applied to help improve model performance. Clustering is a valuable tool for this project when dealing with unsupervised learning models. Since we do not know the features of the data we are not be able to apply domain knowledge to help with finding potential patterns. Instead, we rely heavily on our unsupervised learning approach. We use a supervised learning model such as regression, random forest, neural network, etc. for prediction and our unsupervised learning models help to improve the features used in our final supervised learning model.

The size of the Ubiquant dataset is very large. Our models so far have only been run using samples of the entire dataset.

The Pittsburgh supercomputer will allow us to be able to train our models using much larger sample sizes. We plan on using the supercomputer once we have seen which models seem to be performing the best on the smaller sample sizes and then see if the performance of these models changes significantly when given more data

C. Technical Challenges

One of the primary technical challenges that we face is deciding how to measure market conditions such that we can measure how our approaches perform in each. We must quantitatively define what constitutes different market conditions so that our results are both valid and interpretable.

Another technical challenge that we will face is the ambiguity of the features given to us in the data set. There includes features f-0 through f-299. These metrics remain hidden considering if they were to tell us exactly what these metrics did, then they would be giving away some secrets to their company and process. This will be difficult to figure out the importance of each of these features and which may be more important than others.

Row_ID	Time_ID	Invest_ID	Target	f_0	f_1	f_2
0_1	0	1	-0.300874	0.932572	0.113691	-0.402206
0_2	0	2	-0.231040	0.810801	-0.514115	0.742368
0_6	0	6	0.568807	0.393973	0.615936	0.567806
0_7	0	7	-1.06478	-2.343535	-0.011870	1.874606
0_8	0	8	-0.531940	0.842057	-0.262992	2.330029
0_9	0	9	1.505903	0.608854	1.369304	-0.761515

Fig. 5. Dataset given to us by the Kaggle Competition

Lastly, we will face challenges with creating a model for the 18.55 GB data set given to us without over-fitting and making the model able to adapt to any changes that may occur in the testing data set. Considering the many options for a model, we are going to have to test multiple types of models as well, which will take a lot of time and effort put into each particular model. We don't foresee this as being a huge issue, but it will take time and lots of discussion for our group to reach a final decision.

IV. Experiments

As detailed in our solution framework, we use both supervised and unsupervised learning techniques. Our process involves combining these different techniques in order to obtain an efficacious end model for predicting investments' rates of return from the Ubiquant dataset. In words, the solution framework pipeline entails training, validating, and testing four different supervised learning models on different subsets of the whole dataset in order to determine the best class of model for our task while using unsupervised learning techniques in parallel to identify feature importance. After identifying the best class of model and best features or feature engineering approaches, we use the results to train and validate our final model. We use the Pittsburgh Super Computer to make this process more efficient and to allow us to quickly find the best model architecture. After training and validating, we test our final model on unseen Ubiquant data and obtain Ubiqant results for analysis. The following section details our experiments.

A. Exploratory Data Analysis

We decided to run a few tests on the correlation of the features, considering this would be important to our experiments and our models. If we could figure out some features that give the highest correlation, we could adjust our model accordingly. In figure 6, we see all of the correlations between features.

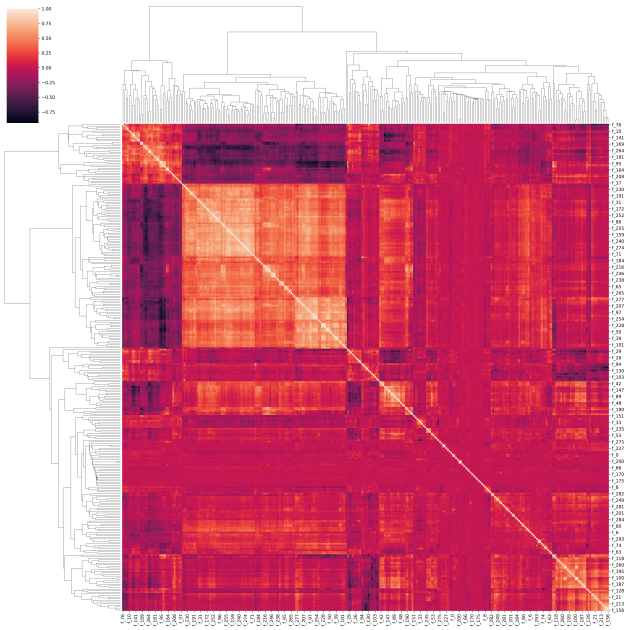

Fig. 6. All intersections of Correlations

We decided to get a better look at these correlations. To do this, we found a few of the top correlations amongst the features and this was insightful because it showed us that a few of the feature numbers were really impactful with other features, and if we were able to, we could impose some sort of pre-existing weights onto the features which would prove to be extremely effective.

B. Supervised Learning

Based on our research the most effective methods for the prediction of returns in financial markets are the deep neural network, recurrent neural network (LSTM), random forest, and ensemble models. We first build models using all of these methods on a common sample from the Ubiqant dataset and analyze the results. We train these models using the first 450,000 observations and test them using the next 50,000 observations of the 3,141,410 total observations in the dataset. The code for our initial models can be found in the GitHub repository [5]. The Pearson Correlation Coefficients of the best models using each of these methods are shown in the table below.

For the Ubiquant dataset we could only use DNN, random forest, and ensemble models. The LSTM model is a recurrent

Feature 1	Feature 2	Correlation
148	205	-0.9367478
95	148	-0.8732327
133	267	-0.8730756
49	205	-0.8717809
88	133	-0.8650543
95	211	-0.8568796
95	254	-0.8501226
95	158	-0.8399703
72	226	-0.8374662
69	130	-0.8267005
95	97	-0.8216310
95	203	-0.8173023
72	148	-0.8151271
72	211	-0.8139968

Fig. 7. Top correlations amongst the features

	DNN	RF	Ensemble
PCC	0.1343	0.0895	0.0885

Fig. 8. Preliminary Model Results - Pearson Correlation Coefficient for the best deep neural network, random forest, and ensemble models.

neural network which has a main benefit of being able to have memory cells to store previous time steps. However, as stated earlier the features of the Ubiquant dataset are not known and there are no date or time features. Because of this, we were not able to have an LSTM model to train on the Ubiquant dataset.

Keeping with many of the State of the Art prior works we used the root mean-squared error metric for measuring training loss and the Pearson Correlation Coefficient (PCC) for evaluating our models. The first class of model that we used were Multi-layer Perceptron (MLP) Deep Neural networks. We built models with a dropout rate of 0.3, a swish activation function, batch normalization, and varying numbers of nodes and layers. The first set of models that we built were trained for only 40 epochs and simply allowed us to get a feel for which model architectures were better or worse for our task. Our results showed that models generally improved as we reduced complexity (meaning less layers and fewer nodes). Using these results we built a second set of models and trained them for 100 epochs. It is worth noting that these models did not fully converge and that we stopped training in the interest of time but will increase epoch length as we hone in on the best model architecture. The average performance of these models was much improved from the first set. The best overall model was from this second set and had an input layer with 32 nodes, four hidden layers with 16 nodes, a single output node, and obtained a PCC of 0.1343. Overall, the performance of the preliminary deep neural network models exceeded our expectations. The top Pearson Correlation Coefficients from the Ubiquant competition on kaggle are in the 0.15 to 0.16 range and we achieved scores in the 0.12 to 0.13 range using only small samples of the total data and rough model hyperparameters. Moving forward we plan to keep this same general architecture to keep the model complexity low in order to avoid over-fitting while still improving its efficacy through more detailed hyperparameter tuning and increased training data.

The other classes of models that we trained and tested were random forest and ensemble models. These models were comparable in their performances with the best models from each class achieving Pearson Correlation Coefficient metrics of 0.089 and 0.088 respectively. These models are substantially inferior to even the average neural network models and took nearly 3 times longer to run given their unfavorable time complexities. Given these results, we did not continue training these classes after the preliminary models. This result is keeping with many prior research studies and upholds the standard that neural networks are superior to other machine learning models for working with time series data in financial markets.

We continue to fine-tune, train, and test a deep neural network model on the entire Ubiqant dataset. The training set consisted of 2,500,000 observations and the test set consisted of approximately 500,000 observations. We kept the general architecture that performed best over our preliminary experiments which had an input layer with 32 nodes, four hidden layers with 16 nodes, and a single output node. We tuned the dropout rate and activation function hyperparameters over the training data for the entire set in order to take advantage of the increased dataset size. We found that the best deep neural network model for the aforementioned architecture was created using a dropout rate of 0.3 and a swish activation function. This model achieved a pearson correlation coefficient score of about 0.12 which would have been a middle of the pack score in the kaggle competition. Furthermore, upon further investigation we discovered some interesting patterns in how the model was making predictions. When the model's prediction for an investment's rate of return was greater than the actual rate of return, an overestimate, our model achieved a significantly improved pearson correlation coefficient of 0.278, as can be seen in the figure below. On the contrary, when the model's prediction for an investment's rate of return was less than the actual rate of return, an underestimate, our model achieved a significantly lower pearson correlation coefficient of 0.019. The proportion of predictions that were overestimates and underestimates were approximately equal with 326,960 overestimates and 314,450 underestimates.

	All Predictions	Prediction > Actual	Prediction < Actual
# Observations	641,410	326,960	314,450
PCC	0.121	0.278	0.019
MSE	0.810	0.638	0.990
MAE	0.597	0.551	0.645

Fig. 9. Overestimate vs Underestimate Evaluation Metrics

C. Implementation on Cyptocurrency Dataset

For the Bitcoin data we introduce the LSTM model. Through our research we have seen the neural networks generally perform the best for financial market predictions.

With the LSTM model we expected greater accuracy than the DNN, random forest, and ensemble models. With no vanishing gradient for the LSTM model our model should be able to make more accurate predictions. For the Bitcoin data we have a number of features that are common for stock or cryptocurrency data. The goal of this model is to learn from these features and predict the next day's closing price. Being able to accurately predict the future closing price is a great advantage for investors so they know whether they should invest, sell, or keep their current crypto. In Figure 10, you can see what the closing prices for Bitcoin looks like for each day.

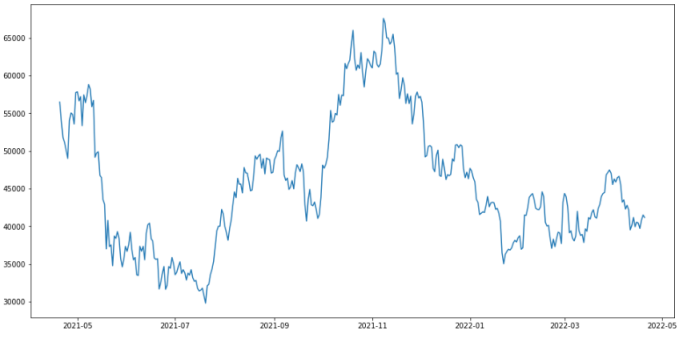

Fig. 10. Line chart of Bitcoin data plotting the closing price from the dataset

Figure 10 helps show the volatility of Bitcoin as closing price is constantly changing. In our model pipeline as you can see below, we decided to use one feature with ten timesteps. Ten timesteps seemed to be really effective and if we went any more less or more, it seemed to make the model worse in most occurrences. We wanted our model to be able to use the long term memory from ten timesteps to keep the changes in mind to give us the best output and guide the customer to the best decision. Just to be clear, our model is not supposed to be used as a cheat code to the stock market. It is merely supposed to provide information on what the stock market may do, and use the previous days to show this. We decided to use three hidden layers with five-ten neurons each. There is definitely a better layer layout which we would like to explore more in the future, but due to time on this project, we stuck with three layers. We could add some normalization layers, dropout layers, and other layers in the future to help make the model even more effective. We also decided to use one hundred epochs in the LSTM model, which proved to be a good choice. We want in the future to also explore the changes that would occur while using a different amount of epochs. This would be an interesting theory to explore more.

First, we convert the dataset to a dataframe, and set the column that contains the date as an index. Then we split the dataset into train and test sets with the train set containing the first 256 rows and the test set containing the rest of the rows so there is about a 70-30 split for the train and test sets. We also normalize the training set to fit values between 0 and 1 and split the training set into x and y training sets. Next, we create the neural network. We experimented with multiple different neural networks to try and find what is the optimal number of layers and nodes to get good accuracy and avoid overfitting.

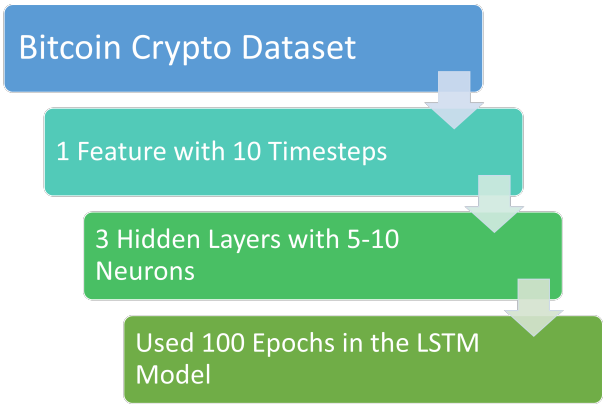

Fig. 11. Shown here is the pipeline for our Cryptocurrency dataset and the specifications for our LSTM model

A problem that can occur with LSTM models is that instead of making predictions the model can just start memorizing the data which can lead to overfitting.

One of the best models in terms of accuracy was an LSTM model with six layers. One input layer with ten nodes, four hidden layers with five nodes, and an output layer. After experimenting with number of epochs it became clear that 100 epochs was a good number. Having more than 100 epochs did not have a noticeable effect on improving the model for any of the LSTM model variations that were made. Having fewer than 100 epochs led to the model predicting a single price for all the predictions. This model had the best accuracy at 97.7006, and will be referred to as Model 1, but this model only tested five days.

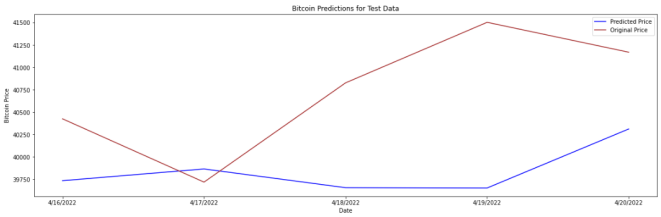

Fig. 12. Line chart plotting the predicted closing price and actual closing price

We then decided to increase the number of days to be tested. We found out after changing number of layers and nodes that the model described above had the best performance for accuracy. The model has One input layer with ten nodes, four hidden layers with five nodes, and an output layer, but is tested for 111 days rather than five. The accuracy for this model became consistently worse as it dropped to 95.7006 and will be referred to as Model 2. The drop in accuracy could be from not having enough training data because of the increased share of the data being used for testing.

Increasing the number of nodes or layers did not improve accuracy. Increasing the number of nodes had similar accuracy as Model 2. Increasing the number of layers started to give worse accuracy than Model 2. Then we decided to decrease the number of testing days to 50. With the decrease in accuracy for all attempted models when using 111 testing days we

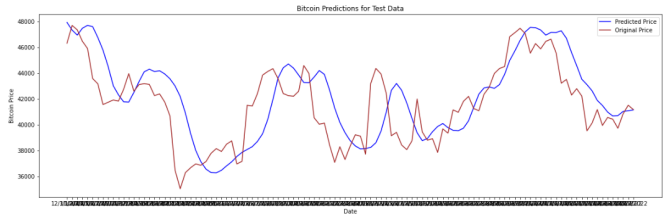

Fig. 13. Line chart plotting the predicted closing price and actual closing price. This chart shows a similar LSTM design as used for Figure 12, but increases the number of testing days which led to worse accuracy.

decided to decrease the testing days to see if performance would increase due to a larger training set. With the decrease in testing days the accuracy did improve to 97.3906 with six layers, 25 nodes for the input layer, six nodes for the hidden layers, and an output layer. This model will be referred to as Model 3. Model 3 performed better than Model 1 by a small margin for this change in testing days. With 50 testing days Model 1 got an accuracy of 96.7974.

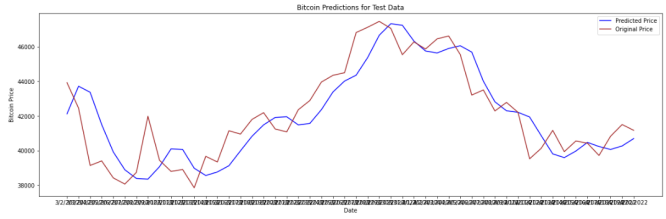

Fig. 14. Line chart plotting the predicted closing price and actual closing price. This chart shows an LSTM design of an input layer with 25 nodes, four additional hidden layers with six nodes each, and an output layer. This model was tested for predicting closing price for 50 days

We decided the best model is Model 3, because it has better accuracy than Model 1 for 50 testing days. Basing the model off of five testing days is not enough to be able to tell if the model can actually perform well. Also, with having so much training data and only five testing days the model runs a strong chance of overfitting. In Figure 15 you can see a Model 3's predictions on closing price for the training and testing data.

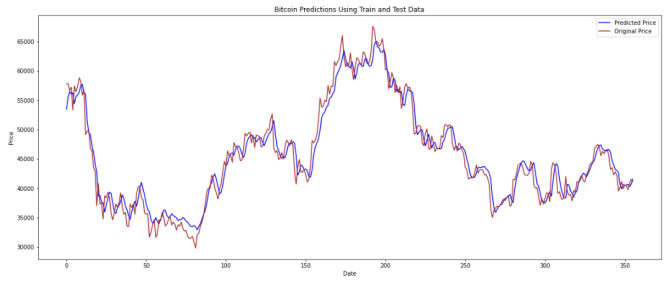

Fig. 15. This line chart shows Model 3 plotting the predicted closing price and actual closing price for both training and testing data.

V. Discussion

Given the time constraints of this project, we were not able to achieve all of our aims. Specifically, we were not able to apply the deep neural network and LSTM models to both the Ubiqant and the cryptocurrency datasets in a way that would have allowed us to make appropriate comparisons. This was largely due to our difficulty of obtaining a cryptocurrency dataset and the difficulty of training and testing models on the extremely large Ubiquant dataset. This prevents us from making definite conclusions about the patterns in performance of different machine learning modeling techniques in the two market contexts. Future work should pick up where we left off such that these conclusions can be made and those with an interest in machine learning as it is applied to financial data may have some useful rules of thumb for model selection. This being said, we made some substantial progress with our research in terms of understanding the Ubiquant and cryptocurrency datasets, understanding how effectively certain models work for each market, and discovering some interesting patterns in the datasets and models.

Independent of the actual research question at hand, this project taught us some valuable lessons regarding how to approach a large-scale machine learning project. One such lesson is that training, validating, and testing machine learning models on large datasets is difficult. Most real world datasets are exceedingly large and it is a necessary skill to understand how to efficiently deal with them. Factors such as time complexity that need not be considered with smaller datasets become of the utmost importance when working with a dataset of a few million observations. We had not had previous experience with a dataset of this magnitude and it ended up being one of the biggest obstacles towards progressing further with the project. Our biggest takeaway here is that the efficacy of a model is not the only thing of importance in a real-world context. Resources are finite and any project with practical implementation must take into account the amount of computation and time required to achieve a goal. Another lesson that we learned is that the conclusions that can be drawn from a data science project involving machine learning are highly dependent on the nature of the study. For example, a project that is investigating a question that has been previously investigated for the purpose of reinforcement or refutation then there is a precedent for the techniques that should be used and their is little room for novel approaches. On the other hand if the goal of the project is novel, like our own, then existing research approaches are largely unhelpful. The main takeaway here is that we should have spent more time carefully formulating our solution as there was not much existing research to base our approach on. Overall, this study made some significant progress towards investigating the effectiveness of various machine learning techniques and for predicting an investment's rate of return and taught us many lessons about how to approach a real-world data science project.

VI. Conclusion

Financial markets are very important in the world for investors and researchers. It has been difficult to make predictions about investments using traditional mathematical techniques. With this project we apply machine learning to try and accurately predict an investment's rate of return. Learning about financial markets and different factors of the market

is essential to create accurate models and to help understand how different factors influence the end models. However, with the features of our dataset being ambiguous it was a major challenge to figure out some of these features and their importance in predictions with different models. The use of unsupervised learning can be used to help with this problem and could help supervised learning models.

LSTM models have generally been some of the best models for predicting financial markets. Our goal with this project is to test this model and others on cryptocurrency data which is a fairly new and lucrative financial market to be able to help investors in buying or selling cryptocurrency.

VII. Team Members Work

Ben: Did majority of Supervised Learning code and helped implement multiple models to give us results on our datasets. Also helped with the creation of many figures and finding related works.

Brian: Completed majority of LSTM model work with the Cryptocurrency dataset, and was the sole worker on gathering the crypto data. Also helped with great figures and pictures of the LSTM work, and helped with related works as well.

Marty: Did majority of correlation work with the features and feature analysis. Helped with presentations and related works as well.

References

[1] Ubiquant market prediction. https://www.kaggle.com/c/ubiquant-market-prediction/data?select=train.csv.

[2] Reaz Chowdhury, M Arifur Rahman, M Sohel Rahman, and MRC Mahdy. An approach to predict and forecast the price of constituents and index of cryptocurrency using machine learning. *Physica A: Statistical Mechanics and its Applications*, 551:124569, 2020.

[3] Guangyu Ding and Liangxi Qin. Study on the prediction of stock price based on the associated network model of lstm. *International Journal of Machine Learning and Cybernetics*, 11(6):1307–1317, 2020.

[4] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[5] Martin Boyle, Ben Jochem, and Brian Kasper. DS340W, 2 2022. https://github.com/17kasperb/DS340W.

[6] Higor YD Sigaki, Matjaž Perc, and Haroldo V Ribeiro. Clustering patterns in efficiency and the coming-of-age of the cryptocurrency market. *Scientific reports*, 9(1):1–9, 2019.

[7] Yulei Rao Wei Bao, Jun Yue. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. (7):1–24.