[COMPARISON OF MACHINE LEARNING METHODS FOR BITCOIN PRICE PREDICTION]

Project Proposal

Information Technology Capstone Project

COMP5703/5707/5708

Group Members

1. Quang Trung Nguyen (470518197)
2. Mingxuan Li (470325230)
3. Bin Liu (460130198)
4. Jun Xiong (470540420)
5. Jiaming Wei (460466754)

ABSTRACT

Bitcoin is a decentralized digital currency which was invented in 2009 and now has become the world's most famous cryptocurrency with millions of daily transactions. In recent years, the price of Bitcoin has attracted a growing amount of attention from the public globally due to its frequent fluctuation. This project aims to build 9 predictive models using traditional machine learning, deep learning, and time-series methods to give a prediction for daily Bitcoin price and make a comparison of all models in order to propose the best price forecasting solution. We collect and use the following data from 2017 and 2018 to build these models for efficiency and accuracy: Bitcoin trading data, stock market indices, foreign exchange rates, social trends and Bitcoin technical indexes. The project also attempts to build visualizations of Bitcoin price prediction and correlation and present them on a public static webpage hosted by Amazon Web Service.

TABLE OF CONTENTS

1. INTRODUCTION

Bitcoin is the most famous cryptocurrency invented by Satoshi Nakamoto, a purely peer-to-peer transaction system and network uses digital signatures to timestamps the transactions with an ongoing chain of hash-based proof-of-work (Nakamoto, 2008). From its creation to now, Bitcoin and other cryptocurrencies are usually identified as economic bubbles or ponzi schemes by the public and its price fluctuates significantly by speculation, FUD (Fear, uncertainty and double) and FOMO (Fear of missing out). The motivation of this project is to compare different methods such as time series, machine learning and deep learning to build an efficient and accurate model to predict future prices of Bitcoin. Moreover, we aim to visualize the correlation between Bitcoin price and factors that have influence on it. The potential beneficiaries are cryptocurrency investors, academic researchers and regulators, the investors can use our trading indicators to make investment choices, academic researchers can use our models and comparisons to do any further analysis.

2. RELATED LITERATURE

Traditional Machine Learning Methods

Several traditional machine learning models have been implemented successfully for Bitcoin price prediction over the years. In general, tree-based models such as Random Forest, Decision Tree and eXtreme Gradient Boosting (XGBoost) and regularised regression model like Elastic Net have been shown to outperform both simple linear regression model as well as time-series models (Guo & Antulov-Fantulin, 2018). In general, tree-based model performs best when predicting short-term Bitcoin price of within 10 days and XGBoost was able to achieve the lowest test error (Alessandrettia et al., 2018). Although in many studies, deep learning techniques such as Recurrent Neural Network and Bayesian Neural Networks outperform traditional machine learning models with lower test errors (Jang & Lee, 2018), they have higher computational costs to train the models which is a disadvantage. Since our training dataset is relatively small, that difference might not be significant. In 2014, Shah and Zhang (2014) applied Bayesian regression algorithms to predict Bitcoin and was able to double their Bitcoin investment in 50 days. However, they used data from 2014 which has fewer fluctuations, so their model might not be as effective on today highly volatile data. That said, it would be interesting to test this model on today data and compare its performance with other machine learning models.

Deep Learning Methods

A feedforward network is the most basic deep learning architecture, but it has no conception of order in time ("A beginner's Guide to LSTMs", n.d.) as it only considers the current input it has been given to, hence it is not suitable for time-series data in our case. Comparing to feedforward network, the recurrent network (RNN) not only considers the current input but also inputs it has been given previously. In his paper, Chung et al. (2014) indicates that RNN is an extension of a conventional feedforward neural network, the activation function of RNN is in a recurrent hidden state that can handle sequence which dependents on that of previous time. RNN with Long Short Term Memory (LSTM) units works well on sequence-based tasks with long-term dependencies. According to a research blog on understanding LSTM networks (Olah, 2015). Traditional neural networks cannot perform like humans because it is unclear how to use its reasoning about previous works. However, RNN addresses this issue because they have loops in the network that allows information to be persisted throughout. LSTM has been applied in several studies for predicting Bitcoin price and it has shown to achieve very

good test errors. One study comparing its performance to RNN using Gated recurrent units (GRU) shows that single-layered GRU outperforms single-layered LSTM in prediction accuracy, however adding more layers to LSTM would improve its performance (Bobriakov, 2018).

Time-Series Methods
In time-series analysis, ARIMA - AutoRegressive Integrated Moving Average (Asteriou & Hall, 2011) is selected to fit to the dataset to better understand the data or to predict future points for forecasting. This method is proposed by Box and Jenkins in the early 1970s, ARIMA models aim to describe the correlations in the data with each other, it can also take into account the seasonality of dataset which is highly suitable to Bitcoin price data.

3. PROJECT PROBLEMS

3.1. Project Objectives

The project aims to (1) implement 9 machine learning methods to predict Bitcoin price and compare them based on prediction accuracy, (2) develop a static webpage that visualizes the prediction results of these methods, and the correlation between Bitcoin price and other variables. The project should cost $0 to implement and should be completed within 13 weeks from 8 August 2018 to 2 November 2018.

3.2. Project Questions

The project addresses the following questions:

- What is the optimal machine learning or statistical model for predicting Bitcoin price? To answer this question, we will train 9 predictive models based on 9 different methods, using daily Bitcoin trading data, and calculate and compare their test errors.
- Does there exist strong correlations between Bitcoin price and other variables such as other cryptocurrency prices, commodity prices, major stock market indices, and social trend data?
  This question will be answered by visualizing these correlations on our static webpage using data from our collected datasets.

3.3. Project Scope

Part 1 - Modelling: We implement and compare the results of the following 9 models to predict daily price of Bitcoin, using data from 2017 and 2018.

- Traditional Machine Learning: (1) Bayesian Regression, (2) Ridge Regression, (3) Lasso Regression, (4) Elastic Net, (5) Random Forest, (6) XGBoost.
- Deep Learning: Two Recurrent Neural Networks: (7) Long Short Term Memory - LSTM, (8) Gated Recurrent Unit - GRU.
- Time-Series: (9) ARIMA

Part 2 - Visualisation: We develop interactive visualizations of the following and deploy them on a static webpage:

- Time-series graphs of Bitcoin price prediction from each of the 9 models.
- Correlation matrix between Bitcoin price and other cryptocurrency prices
- Correlation matrix between Bitcoin price and other variables from the dataset used for prediction, description of these variables can be found in Appendix 1.

4. METHODOLOGIES

In this section, we start by describing in detail the chosen traditional machine learning, deep learning, and time-series methods that we will use to predict the price of Bitcoin. We also explain how we collected our datasets, how we will deploy visualizations on the webpage, and perform testings.

4.1. Methods

4.1.1. Exploratory Data Analysis

The main purpose of data analysis in our project is to study the characteristics of our dataset to understand its underlying structure and also to clean and prepare it for modelling. First, we will handle any missing and null values in the data and obtain descriptive statistics such as count, mean, standard deviation, minimum, maximum, quantiles of the variables. Due to its time-series nature, we will examine if our data is stationary and is affected by trend or seasonality by performing a seasonal decomposition on the data. After that, we will perform autocorrelation check to discover repeating patterns in our data. Next, we will visualize the correlation between all variables to identify any non-linear relationships and multicollinearity, and to discover how correlated each variable is to Bitcoin price. We will also visualize and compare data distribution of each variable. Since our features are highly diverse in scale, we will rescale the features to a given range of 0-1 by using *sklearn.preprocessing.MinMaxScaler*. We will use sklearn, pandas and matplotlib to perform EDA on our dataset.

4.1.2. Traditional Machine Learning Methods

Feature Engineering can enhance the performance of predictive models as this process transforms raw data into meaningful and relevant features as inputs into machine learning models. Moreover, it can prevent our models from overfitting since our datasets have relatively a small sample size (546) compared to the number of features (32). We divide feature engineering into two steps.

Step 1: Feature Extraction. For data transformation, since most of our features are highly skewed, we will use Box-Cox transformations on the data to have a normal distribution which is beneficial as many machine learning models assume normal distribution in the data. This is done using *scipy.stats.boxcox*. After that, we add non-linear features to our data by using *sklearn.preprocessing.PolynomialFeatures* which could improve model performance.

Step 2: Feature Selection. Univariate selection will be used to select the best subset of features based on univariate statistical test scores such as F-scores and p-values. This is done using *sklearn.feature_selection.SelectKBest*. After finalizing the best features subset, they will be used to train our models.

Method 1: Ridge Regression

Linear regression is a basic technique to solve regression problems in machine learning, however it is prone to overfitting issue when dealing with datasets with many features such as ours (James et al., 2013). We will use regularised regression methods to overcome overfitting and also because they can handle multicollinearity in data more effectively. Ridge regression works by minimizing the impacts (coefficients) of irrelevant features to avoid overfitting. It will be implemented using *sklearn.linear_model.Ridge*.

Method 2: Lasso Regression
Lasso regression is another regularised technique to avoid overfitting, however it not just minimises the coefficients of irrelevant features like Ridge regression but sets them to 0 to completely eliminate these features. This method will be implemented using *sklearn.linear_model.Lasso*.

Method 3: Elastic Net
This method can be regarded as the combination of Ridge and Lasso regression as it uses regularised terms of both methods. We choose this method as it was shown to be quite robust to redundant features in data, hence it is less prone to overfitting issues. This will be implemented using *sklearn.linear_model.ElasticNet*.

Method 4: Bayesian Regression
Contrary to regular linear regression method which gives a single estimate of the model parameters, Bayesian Regression outputs a distribution of the estimate which helps us infer the uncertainty of our model. This method was successfully applied to predict short-term Bitcoin price quite well as it automatically discovers important patterns in the data (Shah & Zhan, 2014). We will implement this model using Python *PyMC3* package.

Method 5: Random Forest
Random Forest is an ensemble method that computes multiple decision trees and outputs the mean of predictions from these individual trees (Ali et al., 2012). Advantages of Random Forest include overfitting prevention as it is less sensitive to noises and outliers, and automatic feature importance generation which helps us better understand the features. We will use *sklearn.ensemble.RandomForestRegressor* to implement this model.

Method 6: XGBoost
XGBoost (eXtreme Gradient Boosting) is an implementation of Gradient Boosting Algorithm for regression and classification problems. Its advantages over other tree-based models is computational speed and superior model performance (Chen & He, 2015). Indeed, XGBoost was shown to be the best performing tree-based model in predicting Bitcoin price. Similar to Random Forest, XGBoost also generates feature importance automatically. We will use *xgboost.XGBRegressor* to implement this model.

4.1.3. Deep Learning Methods

We will use Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures of the recurrent neural network (RNN) to demonstrate the prediction of the bitcoin price by neural network. We choose RNN as it has been shown to work well with time-series data and has been previously applied successfully to predict Bitcoin price. We use Keras for training the model and try to fine-tune the following hyper-parameters: different number of layers and units, activation functions, optimizers, epoch and batch size.

Method 7: LSTM
LSTM is a variation of RNN. LSTMs can preserve the error that can be back-propagated through time and layers, which allows current nets to continue to learn over and over again ("A beginner's Guide to LSTMs", n.d.).

Method 8: GRU
A report from Chung et al. (2014) indicates that GRU will make each recurrent unit to adaptively capture dependencies of different time scales. Similarly to a LSTM unit, GRU has gating units that regulate information flow inside the unit without needing separate memory cells.

4.1.4. Time-Series Method

Method 9: ARIMA
We choose ARIMA as our time-series model because it is robust in predicting financial data like Bitcoin price. ARIMA(p,d,q) is the standard notation of the ARIMA method, we plan to use the following steps to perform the ARIMA: (1) Identify the stationarity of the dataset, autocorrelation function and partial autocorrelation function; (2) Smoothing non-stationary time series; (3) Establish the model by calculating the lag - q, p and order of difference - d

4.2. Data Collection

We collected data from multiple sources and combined them into the following datasets in CSV format which have been uploaded to our GitHub repository:

- **bitcoin_train.csv** (546 rows, 33 columns) is our training set which contains historical daily Bitcoin trading data (date, open price, high price, low price, close price, volume, total market cap) and other 32 features including commodity prices, foreign exchange rates, stock market indices, social trend data, and Bitcoin technical data from 01/01/2017 to 30/06/2018. A detailed description of the features can be found in Appendix 1. This dataset will be used to train our models.
- **bitcoin_test.csv** (92 rows, 33 columns), a held-out test set that contains data from 01/07/2018 to 30/09/2018, will be used to measure prediction accuracy of the models.
- **crypto_price.csv** (546 rows, 17 columns) contains historical daily prices of 17 cryptocurrencies with the highest market capitalisation from 01/01/2017 to 30/06/2018 will be used for correlation visualization.

We used the Python package *cryptory (https://github.com/dashee87/cryptory)* which has built-in functions to collect Bitcoin trading data, and for other data we downloaded them manually as CSV files and all data were read into Pandas DataFrames using pandas.to_csv(). We then joined these DataFrames into a single DataFrame on *date* as key using pandas.DataFrame.merge(), and output it into CSV files. The sources of our dataset are included in Section 5.3 - Materials.

4.3. Deployment

In this project we will be using time series, traditional machine learning and deep learning methods to build the models, the models will be built by our specified computing engine which could be either our local environment or other distributed systems. The data and models will be uploaded to GitHub repository, for the visualisation part, we will be providing a deployment script to deploy the visualisation code (d3.js) to Amazon Web Service S3 Buckets publicly to share our result. Amazon S3 (Simple Storage Service) is an online service provided by Amazon.com which allows storing large amount of data and hosts websites (“Amazon EC2”, 2015). D3.js is a JavaScript library for manipulating documents based on data, it combines powerful visualization components and a data-driven approach to DOM manipulation. (Zhu, 2013). We will also be referencing *https://d3js.org* for good visualisation examples to give users better understanding about the insights of price predictions and model comparisons.

4.4. Testing

For all models, we will use 10-fold Cross-Validation to compute training error on the training dataset, and a held-out test dataset will be used to compute their test error. Two metrics, Root Mean Square Error and Mean Absolute Error, will be used. Before the deployment of the project, testing is a necessary phase to assure the functionality, stability and usability of the system. Since the project has two independent sections, that are modelling and visualization, a series of testing methods are applied to identify errors and fix bugs. For coding models using Python and visualization using d3.js, unit testing is used in every single stage to test each individual function of the program to make sure it is running smoothly without any issues. After that, we perform integration testing to test the program as a whole to examine if individual functions interact correctly. For data visualization, the project proposes to build visualizations of Bitcoin price prediction based on public static website hosted by AWS. Functionality testing is utilized to assure the usability of the project which includes (1) Interactive functions of each visualization, (2) How these visualizations are presented via the static web page.

5. RESOURCES

5.1. Hardware & Software

Hardware

We will use our own computers for data analysis, models training, data visualisation and web page deployment. Specifications of our computers: Mac OS Operating System, Processor 3.5 GHz Intel Core i7, Memory 16 GB 2133 MHz LPDDR3.

Software

The following software will be used:

- Anaconda, Python 3.6
- XGBoost library, Keras library, D3.js JavaScript library,
- Amazon Web Services (AWS) & Simple Storage Service (S3),
- Python external libraries: *sklearn, pandas, numpy, statsmodels, scipy, spark-sklearn, PyMC3, matplotlib, cryptory.*

5.2. Data Sources

Data	Sources
Bitcoin Trading Data	Coinmarketcap
Stock Market Indices, Commodity Prices, Foreign Exchange Rates	Yahoo! Finance
Bitcoin Twitter Followers Growth	SocialBlade
Bitcoin Wikipedia Page Views	Wikimedia Toolforge
'Bitcoin' Google search trends	Google Trends
Bitcoin Technical Data	Digiconomist, Blockchain.com

5.3. Roles & Responsibilities

Members	Roles	Responsibilities
Quang Trung Nguyen	+ Machine Learning Modeler + Trouble Shooter	• Data Collection • Machine Learning Models Development & Testing • Develop variables correlation matrices using d3.js • Format and proofread reports • Identify and manage project risks
Mingxuan Li	+ Time-Series Modeler + Front-end Programmer	• Time-Series Models Development & Testing • Develop models prediction results visualization using d3.js • Deploy d3.js codes into Amazon Web Services
Bin Liu	+ Group Leader + Organizer	• Machine Learning Models Development & Testing • Schedule group meetings • Monitor progress of the project
Jun Xiong	+ Deep Learning Modeler + Data Analyst	• Deep Learning Models Development & Testing • Exploratory Data Analysis
Jiaming Wei	+ Deep Learning Modeler + Tester	• Deep Learning Models Development & Testing • Test functionality and usability of the webpage

6. EXPECTED OUTCOMES

6.1. Project Deliverables

No	Deliverables	Expected Completion Date	Notes
1	Prediction Models	Week 8 19/09/2018	Python codes implementing 9 predictive models (Traditional, Time-Series, Deep Learning)
2	Data Visualizations	Week 9 26/10/2018	D3.js codes visualizing: - Time-series graphs of prediction from 9 models - Correlation matrix between Bitcoin price and other variables in the dataset
3	Static AWS Webpage	Week 10 10/10/2018	The AWS hosted static webpage displays the visualizations in deliverable 2

4	Group Proposal Report	Week 5 31/08/2018	*None*
5	Group Progress Report	Week 9 05/10/2018	*None*
6	Group Presentation	Week 12 26/10/2018	*None*
7	Group Final Report	Week 13 02/11/2018	*None*

6.2. Implications

This project will have some implications to the machine learning and deep learning area. Academic researchers who work in this field can make use of our models and comparison to conduct further studies and analysis. In addition, the results of our project can aid those who are interested in cryptocurrency investment in selecting appropriate machine learning models to predict the price of Bitcoin or other cryptocurrencies. It can potentially help cryptocurrency investors to make better trade decisions.

7. MILESTONES & RISK ASSESSMENT

7.1. Milestones

Milestone	Tasks	Reporting	Date
Week-1	**Project start:** 1.Kick-off Session 2. Familiarize with project requirements & group members' background 3. Set up communication channels via Slack, WeChat, Google Drive and Github	Meeting with tutor and group members to review the project	08/08/2018
Week-2	**Analysis stage:** 1. Define project scope 2. Design work plan and tasks allocation 3. Literature review about project topic	*None*	15/08/2018
Week-3	**Analysis stage:** 1. Literature review about suitable methodologies & data 2. Data collection	Meeting with tutor to review the work plan	22/08/2018
Week-4	**Analysis stage:** 1. Finalise datasets and upload to Github 2. Finalise set of machine learning models to implement and the resources needed 2. Proposal report writing	Group meeting to discuss and write proposal report	29/08/2018

Week-5	**Proposal Report Due**	Meeting with the tutor to review the proposal report	31/08/2018
Week-6	**Development stage:** 1. Perform Exploratory Data Analysis on the dataset 2. Perform Feature Engineering to select the best subset of features for model training	None	05/09/2018
Week-7	**Development stage:** 1. Machine Learning models training 2. Time-Series models training 3. Deep Learning models training	None	12/09/2018
Week-8	**Development stage:** 1. Models Evaluation 2. Build data visualizations using d3.js 3. Progress Report Writing	None	19/09/2018
Week-9	**Progress Report Due**	Meeting with tutor to review the progress report	05/10/2018
Week-10	**Deployment & Testing stage:** 1. Web page deployment 2. Unit Testing on each visualization 3. Integration Testing on the entire webpage	Group meeting to test the webpage	10/10/2018
Week-11	**Final Documentation stage:** 1. Final Presentation preparation 2. Final Report writing	Group meeting to prepare presentation and write the final report	17/10/2018
Week-12	**Final Presentation**		26/10/2018
Week-13	**Final Report (thesis)**		02/11/2018

7.2. Risk Assessment

We present our Risk Register Plan in Table 1. which identifies and categorises the major risks that our project could face. Our Risk Mitigation Plan in Table 2. describe possible responses that we will undertake to handle these risks.

Table 1. Risk Register Plan

Risk No	Category	Risk Description	Probability of Occurring	Impact

1	Data Collection	Python package *cryotory* becomes unavailable to collect data for test dataset	Medium	Low
2	Deployment	Non-indicative visualisations on the webpage	Low	Medium
3	Development	Python external libraries become unavailable	Low	Medium
4	Project Planning	Over-engineering - Implement changes that increase the project scope significantly	Low	High
5	Project Planning	Failure to achieve planned milestones due to time overrun	Low	High
6	Data Collection	The collected datasets contain irrelevant/ incorrect features that can negatively impact machine learning models performance	High	Low

Table 2. Risk Mitigation Plan

Risk No	Response Type	Response	Responsible Persons
1	Mitigation	Manually collect and combine data from multiple sources Store the existing dataset into github repository	Quang Trung Nguyen Mingxuan Li
2	Avoidance	Carefully evaluate different visualisation methods from d3.js library	Mingxuan Li
3	Mitigation	- Implement the models algorithms manually in Python - Use R or other programming languages to implement the models	All members
4	Avoidance	- Follow the planned milestones and methodologies precisely - Hold frequent group meetings to discuss any possible changes made to the project scope	All members
5	Avoidance/ Mitigation	- Monitor project progress frequently and identify any issues early - Reduce the scope of the project, complete all baseline models first before adding complexities and extra features	All members
6	Avoidance	Perform Exploratory Data Analysis and Feature Engineering to eliminate irrelevant features	Quang Trung Nguyen Jun Xiong

REFERENCES

Alessandrettia, L., ElBahrawya, A., Aielloc, L., M., & Baronchellia, A. (2018). Machine Learning the Cryptocurrency Market. *Cornell University Library*.

Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random Forests and Decision Trees. *IJCSI International Journal of Computer Science Issues*, 9, 272-278.

A beginner's Guide to LSTMs. (n.d.). Retrieved from https://skymind.ai/wiki/lstm#long.

Amazon EC2. (2015). Amazon Web Services. Retrieved from https://aws.amazon.com/es/ec2/

Asteriou, D., Hall, S., G. (2011). ARIMA Models and the Box–Jenkins Methodology. *Applied Econometrics*, 265–286.

Bobriakov, I. (2018, March 6). Bitcoin price forecasting with deep learning algorithm. Retrieved from https://medium.com/activewizards-machine-learning-company/bitcoin-price-forecasting-with-deep-learning-algorithms-eb578a2387a3.

Chohan, U., W. (2017). Cryptocurrencies: A Brief Thematic Review. *Economics of Networks Journal.*

Chen, T., & He, T. (2015). xgboost: eXtreme Gradient Boosting.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *Cornell University Library*.

Guo, T., & Antulov-Fantulin, N. (2018). An experimental study of Bitcoin fluctuation using machine learning methods. *Proceedings of ACM Conference* (Conference'18).

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning - with Applications in R*. New York, NY: Springer-Verlag.

Jang, H., & Lee, J. (2018). An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information. *Institute of Electrical and Electronics Engineers*, 6, 5427-5437

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Bitcoin. Retrieved from https://bitcoin.org/bitcoin.pdf.

Olah, C. (2015, August 27), Understanding LSTM Networks [Blog post]. Retrieved from http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Schut, M. (2017). Bitcoin analysis from an investor's perspective. *Erasmus University Rotterdam, Erasmus School of Economics*.

Shah, D., & Zhang, K. (2014). Bayesian regression and Bitcoin. *Institute of Electrical and Electronics Engineers*, 409-414

Zhu, N., Q. (2013). *Data Visualization with D3.js Cookbook*. Birmingham, England: Packt Publishing.

APPENDIX 1 – Dataset Features Description

No	Features	Description	Value Type
1	date	Date	Datetime
2	open	Bitcoin daily opening price (USD)	Float
3	high	Bitcoin daily highest price (USD)	Float
4	low	Bitcoin daily lowest price (USD)	Float
5	close	Bitcoin daily closing price (USD)	Float
6	volume	Amount of Bitcoin exchanged (USD)	Float
7	marketcap	Bitcoin total market value (USD)	Float
8	google_trends	'Bitcoin' Google search interest measure	Float
9	wiki_page_views	Bitcoin Wikipedia page views	Int
10	twitter_followers_growth	Bitcoin Twitter followers gained	Int
11	gold	Gold price (USD)	Float
12	silver	Silver price (USD)	Float
13	platinum	Platinum price (USD)	Float
14	palladium	Palladium price (USD)	Float
15	oil	Oil price (USD)	Float
16	usd_eur	USD to EUR exchange rate	Float
17	usd_jpy	USD to JPY exchange rate	Float
18	usd_gbp	USD to GBP exchange rate	Float
19	usd_cny	USD to CNY exchange rate	Float
20	SP500	U.S. stock market index	Float
21	Dow_Jones	U.S. Industrial stock market index	Float
22	Nasdaq	U.S. stock market index	Float
23	Russell_2000	U.S. stock market index	Float
24	FTSE_100	London Stock Exchange index	Float
25	Nikkei	Tokyo Stock Exchange index	Float
26	SSE	Shanghai Stock Exchange index	Float
27	VIX	Market Volatility index	Float
28	NVDA	Nvidia Corporation stock index	Float
29	GOOG	Alphabet Inc. (Google) stock index	Float
30	hash_rate	Measure of a Bitcoin miner's performance	Float
31	miners_revenue	Total value of coinbase block rewards and transaction fees paid to miners	Float
32	avg_block_size	Bitcoin average block size (MB)	Float
33	difficulty	Measure of how difficult to find a new block	Float
34	est_transc_volume	Total estimated value of transactions on the Bitcoin blockchain (USD)	Float
35	num_unique_address	Number of unique addresses used on the Bitcoin blockchain	Int
36	BECI	Bitcoin Energy Consumption Index	Float

APPENDIX 2 – Project Gannt Chart