Domain Background

Ethereum is a decentralized digital cryptocurrency platform invented by Vitalik Buterin in 2013. It uses public key cryptography and a consensus algorithm called "proof of work" (POW) to prevent denial of service attacks on the system. **[1]** Ether, Ethereum's currency, does not go through a bank, rather, Ether are managed in peer to peer transactions. Therefore Ether transactions are much cheaper.

Transactions are tracked on the blockchain, a distributed ledger. As **Figure 1** indicates, every block in the chain contains a pointer from the previous block, a timestamp, a nonce, and a copy of valid transaction records within that block.

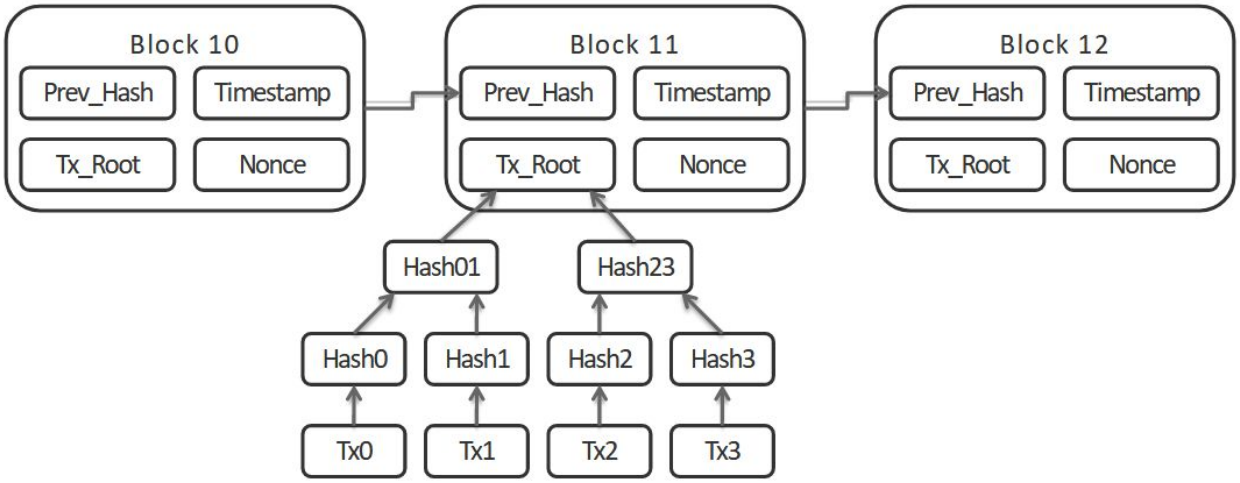

Figure 1 [2]

When a person wants to send Ether to another user, that person will request to have their transaction be validated by the miners and added to the chain.[3] Each miner maintains their own copy of the blockchain, so that they can verify each other's work. Miners race to validate these requests. The miner with the longest chain and a majority approval from the other miners working on their own copies of the chain receives a monetary reward in Ether for the transactions that the miner verified. Only when the miners are in agreement, does the transaction get added to the latest block in their copy of the chain.

Because blockchain is peer to peer and relies on proof of work, it deters people from attempting to alter the history of transactions. This is because a crook would have to fool the other miners that their copy of the chain with an altered beginning transaction is valid. Remember, the miner who has the longest verified chain has their chain accepted as the source of truth for all the other miners. Therefore, a crook would have to do expensive calculations from the early block to the current block faster than all the other miners compute the current block as shown in **Figure 2**.

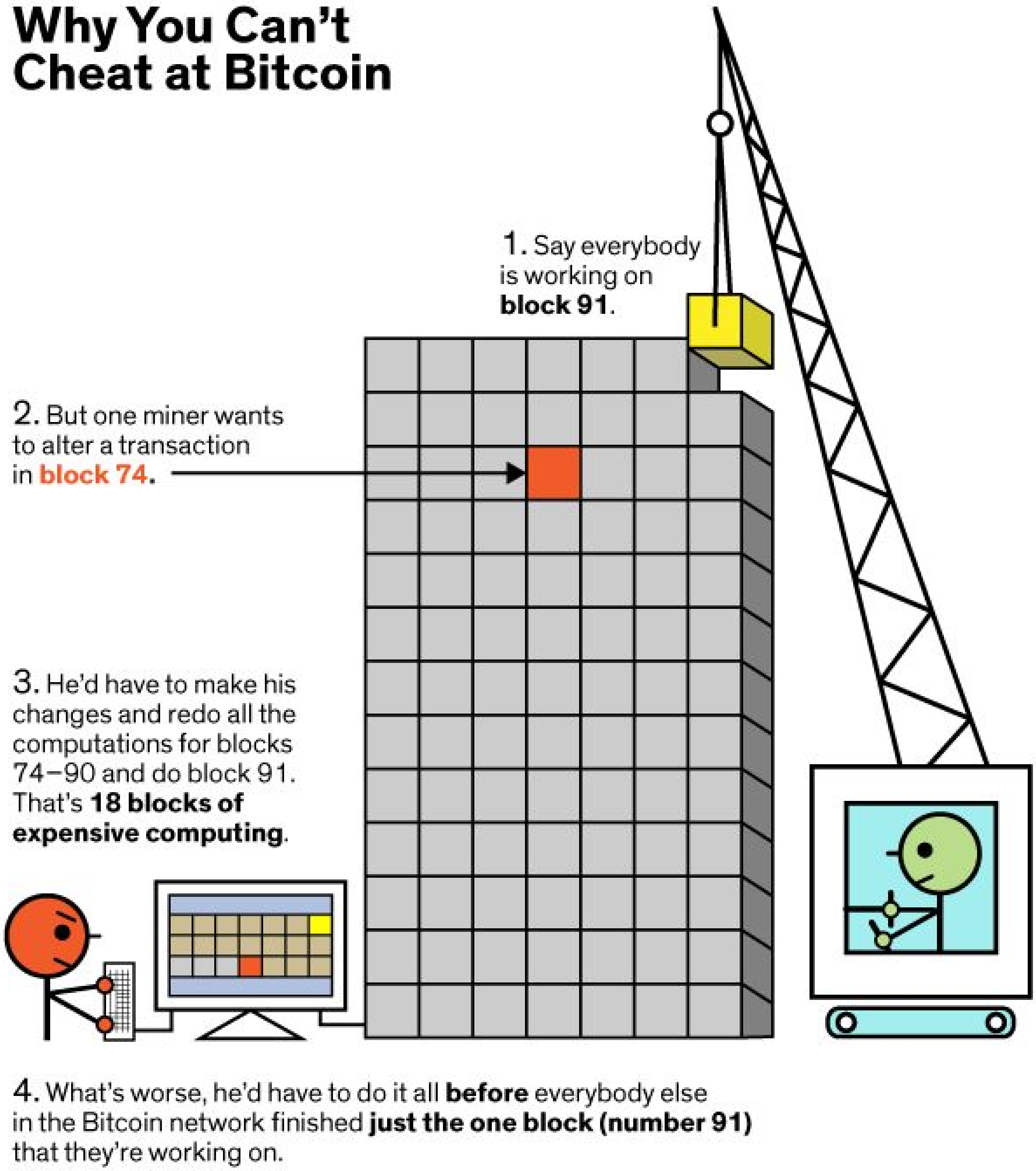
Why You Can't Cheat at Bitcoin
1. Say everybody is working on **block 91**.
2. But one miner wants to alter a transaction in **block 74.**
3. He'd have to make his changes and redo all the computations for blocks 74–90 and do block 91. That's **18 blocks of expensive computing**.
4. What's worse, he'd have to do it all **before** everybody else in the Bitcoin network finished **just the one block (number 91)** that they're working on.

Figure 2 [3]

The more people that use Ether, the more valuable it is, as increased use validates its acceptance in the community. You can see this effect in **Figure 3**, as the number of transactions of Ether have increased over time, with the largest number of transactions, 546,837, occurring on Friday, October 20, 2017. This increased transaction count is correlated to the price of Ether as evident when comparing the price of Ether in **Figure 4** with its transaction count in **Figure 3**.

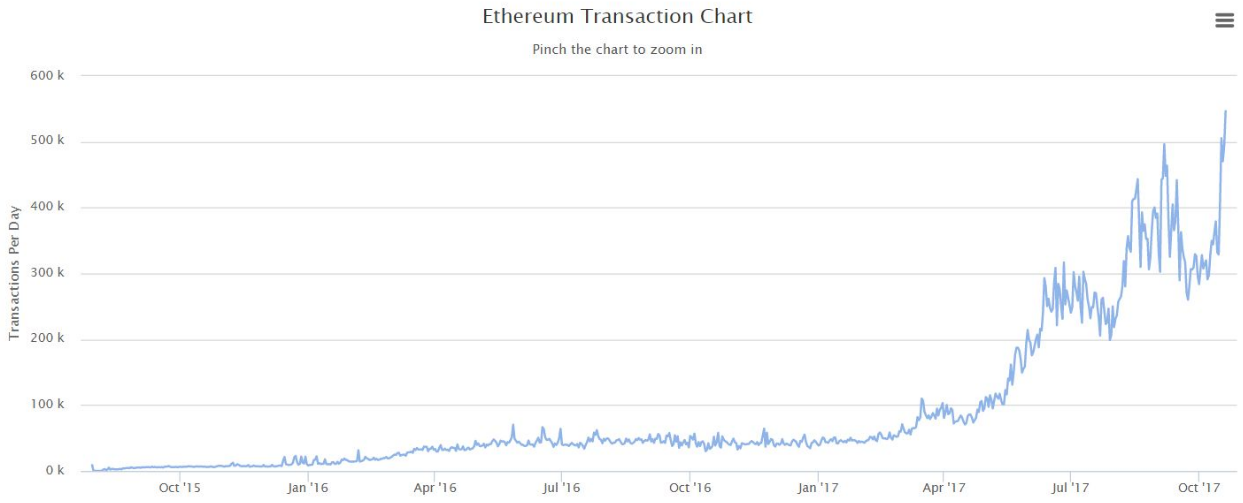

Figure 3 [4]

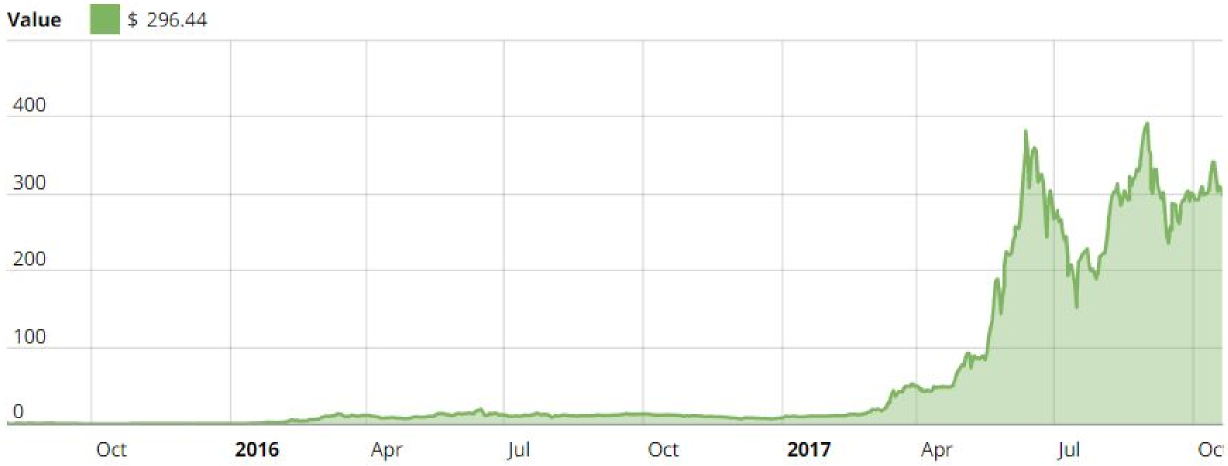

Figure 4 [5]

I want to analyze the price of Ether over time because it is gaining popularity and I personally believe in this product.

Problem Statement

To predict the daily closing prices of Ether for a week into the future. This can be solved with supervised learning since at least a year of data exists with several labels like address, block size, ether price, etc. In addition, I would like to use github contribution history to see if more contributions also influence the price, as I expect more frequent, larger contributions would increase Ether price as the Ethereum framework gets more robust.

Datasets and Inputs

The Ethereum dataset, ethereum_dataset.csv**,** [6] is time series data distributed on Kaggle. The data set was generated by pulling data from Etherscan, which provides metrics about Ethereum. This Kaggle dataset includes transaction date, ether price, gas price, number of transactions on a given day and several others. I would expect that a high price of ether would be correlated to a high number of ether transactions and a lower ether gas price. These features and others will be used to predict the price of ether.

In addition to the Ethereum dataset from Kaggle, I would like to pull data from the top 3 repositories that contribute to the Ethereum framework using Github's GraphQL API https://developer.github.com/v4/. Namely, geth the Go Ethereum client https://github.com/ethereum/go-ethereum, parity the Rust Ethereum client https://github.com/paritytech/parity, and cpp-ethereum the C++ Ethereum client https://github.com/ethereum/cpp-ethereum. I would like to consider the aggregated number of commits of the top three repositories per day. I expect that more commit activity would imply a higher price of Ethereum.

Given both Etherscan Ether data and Github commit data, I would like to focus on the time period from March 2017 to present, since this was the period that Ethereum started to gain popularity. **Figure 5** shows the commit history for go-ethereum over the past year, with the greyed out section being the portion of time I am going to be studying for influence on Ether price. Before March 2017, Ethereum was too unknown, so I do not think that the commit counts would affect Ether price in that time period.

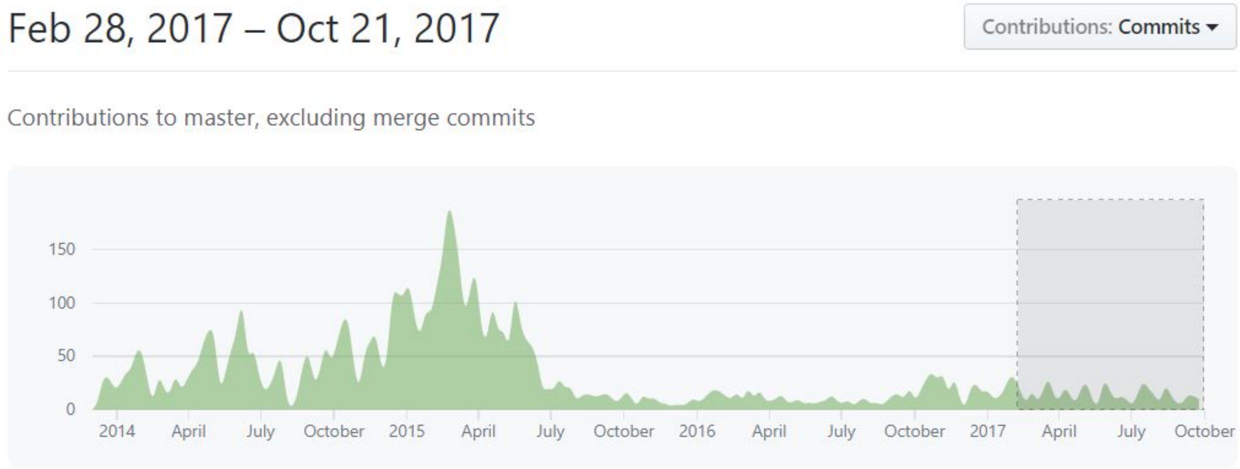

Figure 5 [7]

For instance, looking at one example of the go-ethereum repository's top contributor commits over August 2017 to October 2017 and the price of Ether over these dates, it seems like there could be some correlation. Right around September 17, both commits in **Figure 6** and price of Ether in **Figure 7** dipped.

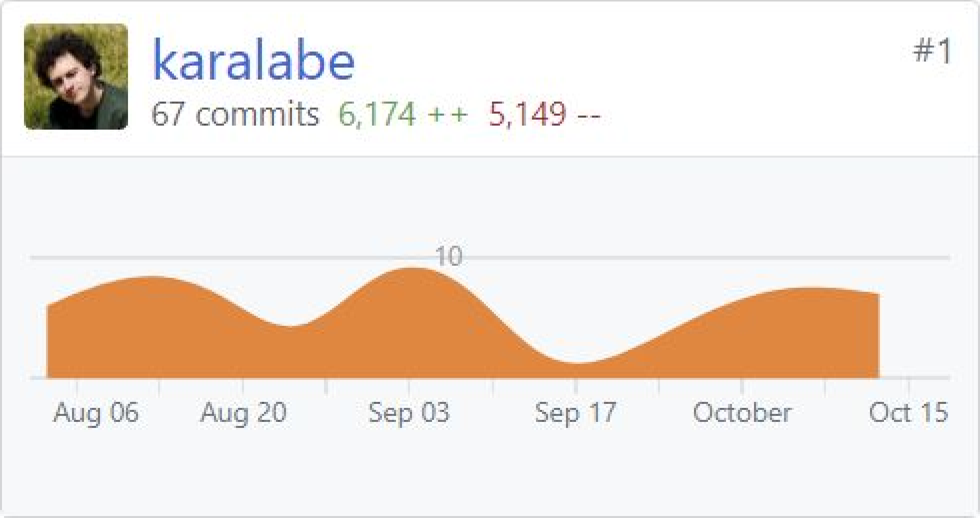

Figure 6 [7]

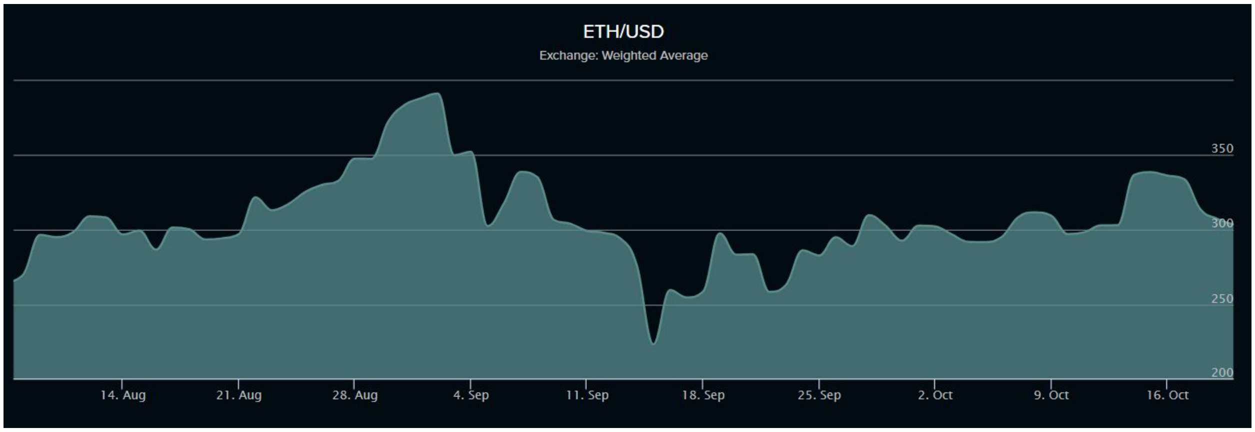

Figure 7 [8]

However, given all these features I described above, I need to reduce the number of features so that the model does not suffer from the curse of dimensionality, i.e. as the number of features grows, the amount of data needed to generalize accurately grows exponentially. I would like to use PCA to identify principal components and reduce noise.

Solution Statement

I would like to start by predicting Ether price over time using a bayesian linear regression model. Bayesian linear regression takes a moderate amount of time to train, so it would be a suitable model to start with to get a general idea of the data. [9] It is a stricter model that would protect better against overfitting and not follow noise as closely.

When choosing this model, I considered the bias-variance tradeoff. The optimal model is one that has low bias, so that the model is complex enough draw relations between features and outputs, and low variance, so that the model can generalize well to new data and not overfit as shown in **Figure 8**.

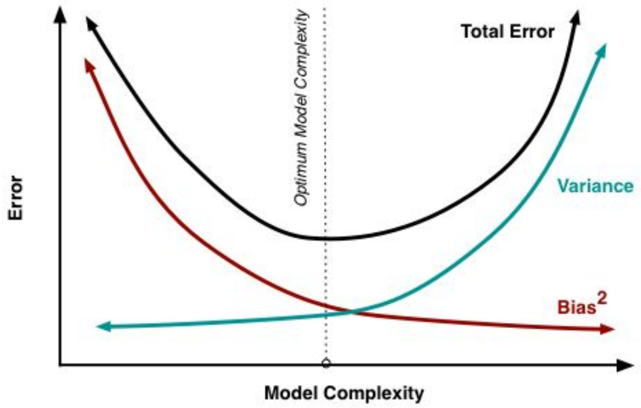

Figure 8 [10]

When deciding on a model, I needed to consider the tradeoff between interpretability and flexibility. The more flexible the model, the harder it is to interpret and the more the model can be interpreted, the lesser its flexibility (**Figure 9**).

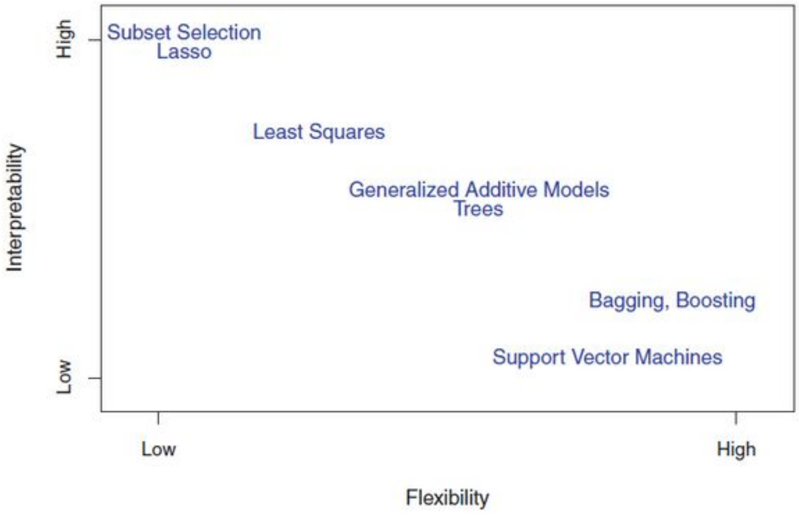

Figure 9 [11]

In addition to a linear model, I would also like to consider using a Long Short-Term (LSTM) Time-Series recurrent neural network (RNN) and also train with an ARIMA model to compare the performance of the three models.

Benchmark Model

A team of MIT students used bayesian regression to predict every two seconds, the average price change of Bitcoin over the next 10 seconds. The team was successful and "over 50 days, the team's 2,872 trades gave them an 89 percent return on investment." [12] They wrote a paper on their Bayesian regression method and how effectively they predicted the price of Bitcoin, a cryptocurrency. [13]

Evaluation Metrics

The accuracy of bayesian linear regression can be measured by its mean squared error, the average squared distance between actual output and prediction:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\widehat{Y}i - Yi)^2$$

Project Design

First, I would create a python script to extract commits over the time period that I have from the Kaggle data. Then, I would merge the two datasets into a features pandas dataframe, extract data from March 2017 onwards, preprocess the data by filling in empty values, normalizing the data, and using PCA to determine what features are most relevant to my predictions. After determining relevant features to use in my predictions, I would also create a outputs pandas dataframe containing the prices of Ether for the 7 days after my features dates. Then, I would divide the data into training and testing data. I would not shuffle the data since I am working with time series data so I will use TimeSeriesSplit. Then, I would train with bayesian linear regression, scikit learn's Bayesian Ridge regression. I would tune the hyperparameters. I would predict the price for the next 7 days. Lastly, I will print the results and the mean squared error as my evaluation metric.

References

[1] https://github.com/ethereum/wiki/wiki/White-Paper

[2] https://en.wikipedia.org/wiki/Blockchain

[3] https://spectrum.ieee.org/computing/networks/the-future-of-the-web-looks-a-lot-like-bitcoin

[4] https://etherscan.io/chart/tx

[5] https://cointelegraph.com/ethereum-price-index

[6] https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory/data

[7] https://github.com/ethereum/go-ethereum/graphs/contributors?from=2017-08-04&to=2017-10-21&type=c

[8] https://ethereumprice.org/

[9] https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-choice

[10] http://scott.fortmann-roe.com/docs/BiasVariance.html

[11] http://www-bcf.usc.edu/~gareth/ISL/

[12] http://news.mit.edu/2014/mit-computer-scientists-can-predict-price-bitcoin

[13] https://dspace.mit.edu/handle/1721.1/101044