Understand Bitcoin: Mining Software

December 8, 2015

Charlie Hume

Dustin Gay

Roma Koulikov

Introduction

Bitcoin (BTC), a digital currency invented in 2009, has a chance of completely transforming the global financial system. There are several characteristics that make bitcoin completely unlike any existing currency. All BTC transactions since the beginning of the currency are logged in an electronic record called a blockchain. Unlike most other currencies, BTC is not controlled by any government. Instead, the BTC algorithm is written with internal controls on the amount of BTC that can be produced. For a regular currency, the overwhelming majority of new money creation occurs with banks lending money into existence through a process known as fractional reserve banking. BTC, in contrast, are mined by anyone willing to devote computing resources to solve a computationally-intensive puzzle. BTC are completely digital. In fact, what we think of as BTC is actually the blockchain record of how many units of the currency were transferred from one party to another or mined into existence. Because of this digital nature, BTC are especially efficient for internet commerce, as the absence of a third-party means that electronics transactions between buyer and seller can occur free-of-charge. The buyer can choose to pays a set fee (less than 40 cents) for the transaction of any size to encourage pools to include transaction in the block that is being mined.

Although the trading volume of BTC pales in comparison to major world currencies, the features described above have made it a popular medium of exchange for internet commerce and investment. One of the integral components of the system, as mentioned above, is bitcoin mining.

Because of the open-source nature of BTC, some of the software that is designed to solve the puzzle to mine new bitcoin is accessible on repositories like Github. In this project, we seek to understand the evolution of Bitcoin mining software. Specifically, we would like to understand the effect that the bitcoin price has on specific types of changes to the mining software.

Bitcoin Miner Investigation and Explanation

In exploring different mining software, we referred to the the record of total commits associated with each miner on Github, the Bitcoin Wikipedia page (https://en.bitcoin.it/wiki/Mining_software) as well as made inquiries on a popular BTC forum (bitcointalk.org) about the most widely-used mining software. From these inquiries, we decided to focus on CGMiner and BFGMiner. We were not successful in obtaining an actual statistic on the percentage of mining transactions attributed to these two miners. However, it appears that numerous pools and BTC-specific hardware use one of these mining tools.

The main purpose of bitcoin mining software like cgminer and BFGMiner is to receive transactions from the bitcoin network, arrange them in a valid block and then send the block’s unique proof-of-work problem to a piece of hardware to solve. Besides that, other features consist of device compatibility, extensive diagnostic information about the mining process, as well as security and stability failover protocols for power outages and network issues. There are also API features for access and remote control.

Hardware

Over time, mining software has evolved to accommodate hardware advancements. When BTC started, mining with a regular CPU was viable. Network traffic was small and the low number of participants allowed for a low complexity proof-of-work. CPU mining quickly became obsolete, however, due to the onset of GPU mining and the resultant rise in network traffic. Because they are exceptionally good at repetitive tasks, GPUs can run about 3200 32-bit instructions per clock cycle, a far cry from the CPU's 4 instructions per cycle. The beginning of GPU mining made CPU mining impractical as the network hashrate grew to such a degree that the amount of bitcoins created by CPU mining became lower than the cost to operate a CPU. The option was therefore removed from the core Bitcoin client's user interface.

In search of even lower running costs, bitcoin enthusiasts began using FPGA mining setups. A Field Programmable Gate Array is basically a circuit board with malleable connections. Operators can design these connections to complete specific tasks as efficiently as the hardware allows, which in the case of bitcoin mining is just a little faster than using a GPU. The advantage of using a FPGA is not just the moderate increase in hashrate, but the decreased power consumption and cost. FPGAs paved the way for even faster mining.

Instead of turning a programmable device into an efficient specialized circuit to produce hashes, why not start with the circuit we want and just build it as a solid microchip? Application-Specific Integrated Circuits (ASICs) are solid silicon microchips, and many exist specifically for bitcoin mining. The rigid structure allows ASICs to run at speeds faster than 100x that of FPGAs, and at a fraction of the cost. ASICs have increased the network hashrate so significantly, that what happened to CPU mining is happening to GPU mining.

Proof-of-work

The overarching purpose of BTC mining hardware is computing a hash using SHA-256. First, the mining software constructs a block of any number of transactions with a block header. This header contains the following static width fields: 4 bytes for version, 32 bytes for the hash of the previous block, 32 bytes for a special hash called a Merkle Root, and 4 bytes for each of the time stamp, number of required 0 bits, and the nonce. The Merkle Root is a special hash value computed by taking the list of transactions in the block, computing the hash of each, then combining two by two and taking the hash again, recursively until there is only one hash value. This value is virtually unique to each list of transactions because the first transaction in every list, called the coinbase, contains a unique bitcoin address to which is sent your earned proof-of-work subsidy.

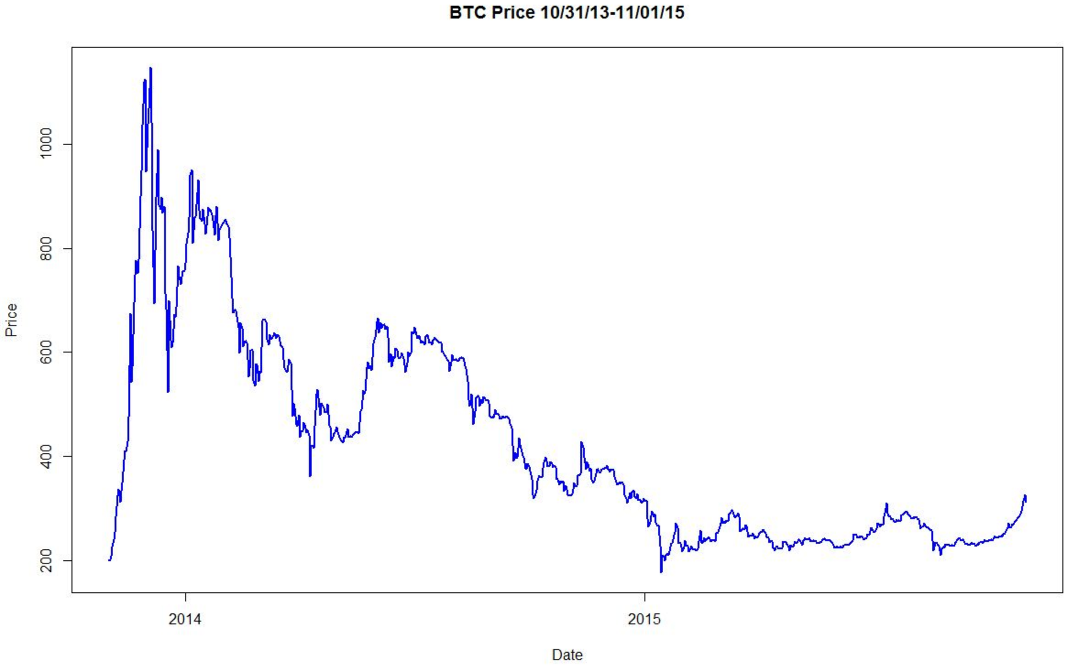
The proof-of-work here, is that the hash of all the fields of the block header concatenated together must have a specified number of the ending bits equal to zero. The number of zeroes required can be fine-tuned to change the difficulty and in effect the success rate of all participants. The nonce is enumerated over all possible values until the hash meets this requirement. If after all enumerations of the nonce, the hash is still not correct, a portion of the coin-base is changed, called the extra-nonce, and the Merkle-Root is recalculated. The regular nonce can then start at 0 again and continue enumerating. Once a valid block is calculated, the hardware returns the correct nonce, and the block can be added to the chain, removing the transactions inside from the global pool. The coinbase transaction, included in the block, is what pays you for successfully finding a valid block containing new transactions that no one else has added to the chain first.

Data Acquisition

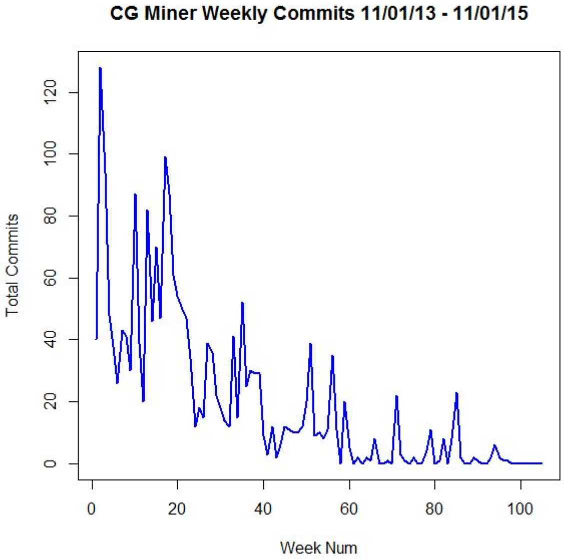
The data used for the following analysis was taken from the github repositories for the mining software. We pulled the commit data for all of the official branches. We used the github api and python to load the responses into a mongodb. We kept a list of hash values as they were inserted into the database make sure each commit was only added once. This was necessary because the earlier commit data would appear in most of the branches and skew the results towards early development. There were a total of 13720 commits from BFG miner and 7655 commits from their entire history and only analyzed the commits between November 2013 and November 2015.

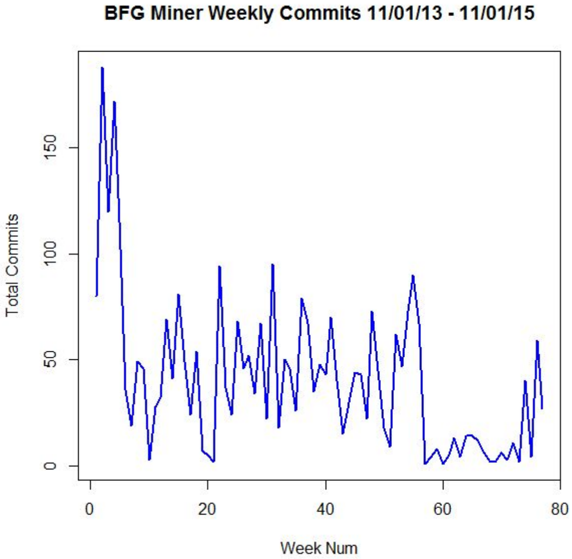
Data Exploration

The chart below shows the BTC price for the past 2 years.



We examine the number of daily commits throughout this same time period.





The numer of total commits associated with each miner seems to be correlated with the BTC price.

The table below shows descriptive statistics for weekly commits for each miner:

	CG Miner	BFG Miner
Mean	19	41

Median	10	36
St. Dev	25.6	37
Max	128	188
Total	2087	3157

Miner Commit Clustering

K Means Clustering

We decided to use the k-means clustering algorithm to assign a commit type to each commit. Using the R tm package, we first created a document term matrix. We then converted the document term matrix to a term frequency inverse document frequency (tfxidf) matrix. The tfxidf matrix assigns weights to words in a document according to their frequency of appearance, and then multiplies that weight by the inverse document frequency (which is 1 if the word appears in all documents, and higher if it doesn't). This measure acts to balance frequent words with those that appear in all documents and thus provide no useful information.

We proceeded to run a k-means algorithm on the tfxidf with 5-10 clusters. To determine the main themes of each cluster, we examined the top 5 words by total weight (summed across all documents in that cluster). Below are the top-5 words for a 6-cluster model:

```
> freq_terms
[[1]]
protocol   header  updated  ability abstract
24.43869 12.58055 11.35892  0.00000  0.00000

[[2]]
   readme   updates     minor    btcsig   options
58.615838 14.538588  6.944583  5.016023  4.881363

[[3]]
      fix  warnings baboptions      typo       ava
 111.32422   29.74164   20.06409   17.89849   11.58372

[[4]]
    style    police     minor   ability  abstract
20.772884  7.526705  2.314861  0.000000  0.000000

[[5]]
                  merge                 version
              238.48306               146.01433
githubcomckolivascgminer                    bump
              138.57027               121.66874
                   work
               85.37105

[[6]]
```

```
     news    release    updates    ability   abstract
228.477178   4.223542   3.355059   0.000000   0.000000
```

For the 6-cluster model above, 90% of the documents were classified into cluster 5. The classification performed in a similar fashion for other k-means models we built. Upon examination of the documents in each cluster, we observed comments with different themes mixed in.

Naive Bayes Classification

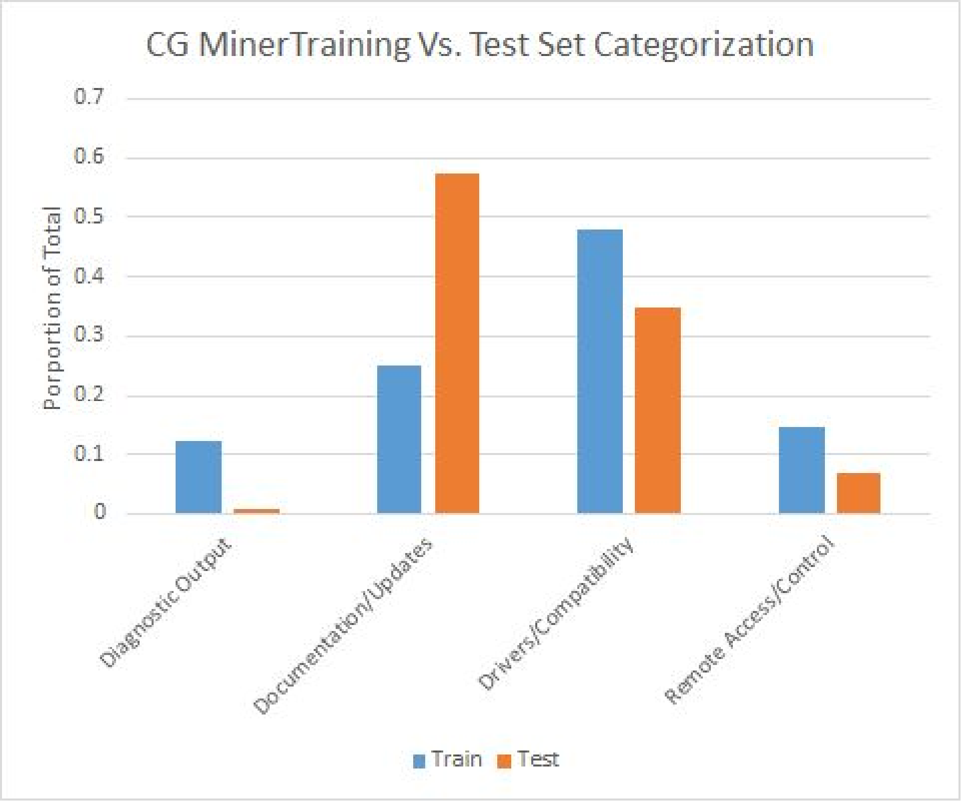
In light of this dissatisfaction with the k-means algorithm, we used a Naive Bayes model for classification. In investigating miners, we noticed 4 major categories of commits: documentation/updates, drivers/compatibility, remote access/control and diagnostic output. For each miner, we manually categorized a training set of approximately 100 randomly-selected comments. We then trained a Naive Bayes model in R and deployed it on the remaining rows in the dataset.

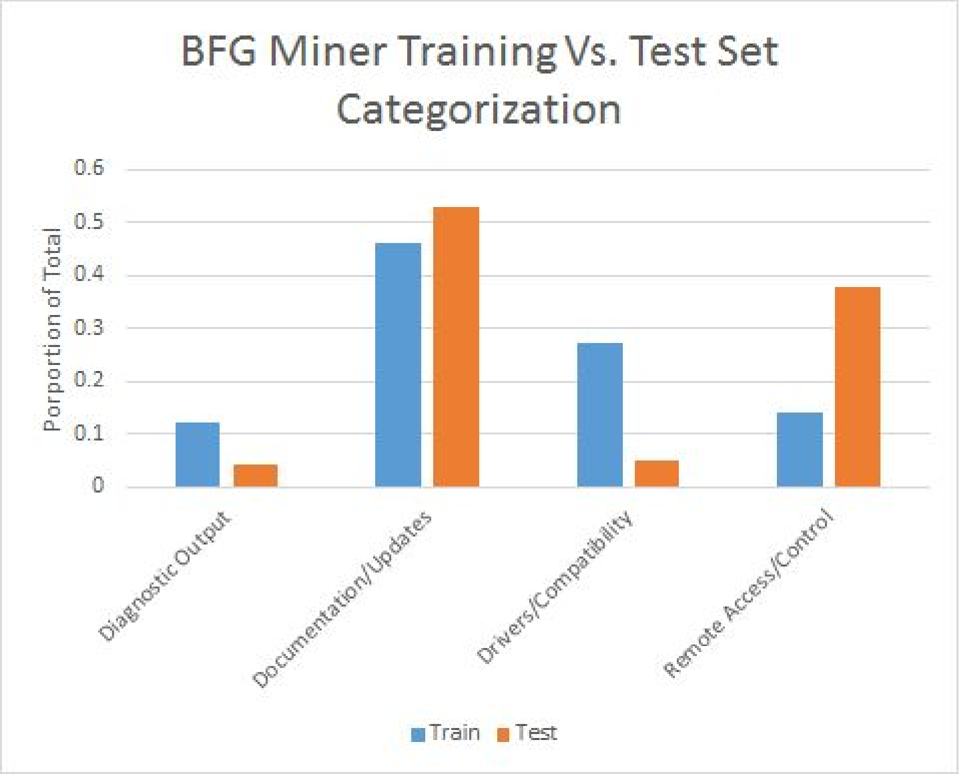
Unfortunately, the Naive Bayes model did a poor job of classifying the documents. For example, while we observed that 46% of the comments in the training set for BFG Miner fell into the Documentation/Updates category, only 1.5% of the documents in the testing set were classified with this category. Similar large discrepancies were observed for CG Miner.

K Nearest Neighbor Classification

In light of the above, we deployed the kNN algorithm on the same training sets for both CG Miner and GUI Miner. The K Nearest Neighbor algorithm plots points (in our case commit comments) from both the training and testing set in an n-dimensional space, where n equals the number of words across all the documents after text pre-processing. For each comment, the algorithm takes the k closest points that are already labeled from the training set. The majority category of these k-closest points is then assigned to the comment. For simplicity, we used a k of 5 in our algorithm. To optimize performance, the kNN can also be trained with various degrees of k.

To test the validity of the algorithm, we compared the proportion of comments in the training and test sets for both miners, as the following charts illustrate:



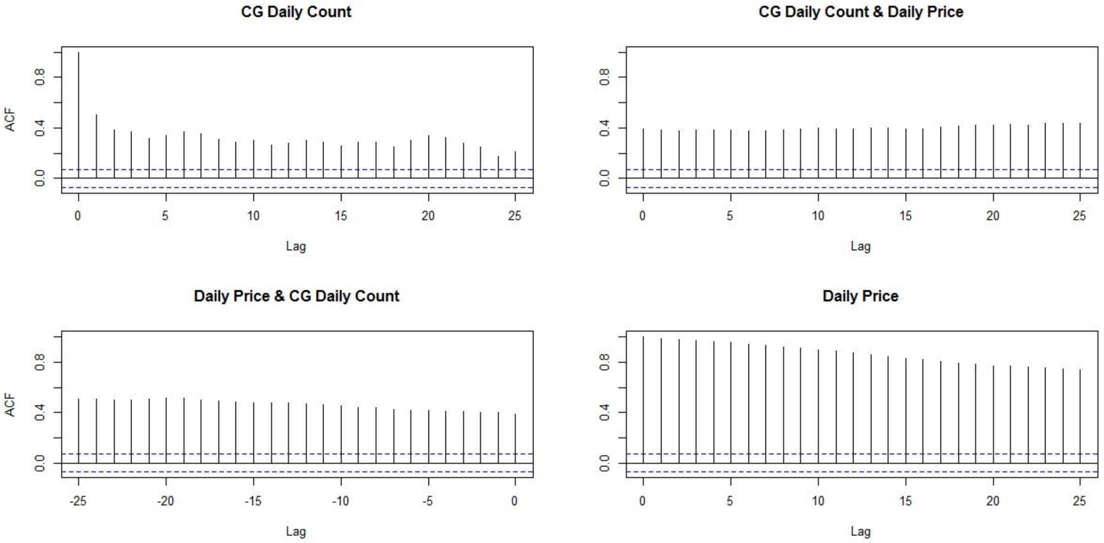


There is a large discrepancy between the training and test sets in the Diagnostics Output categorization for CG Miner. For BFG Miner, there is a large discrepancy in the Drivers/Compatibility categorization. The remaining categories are satisfactory.

Time Series Analysis

CG Miner

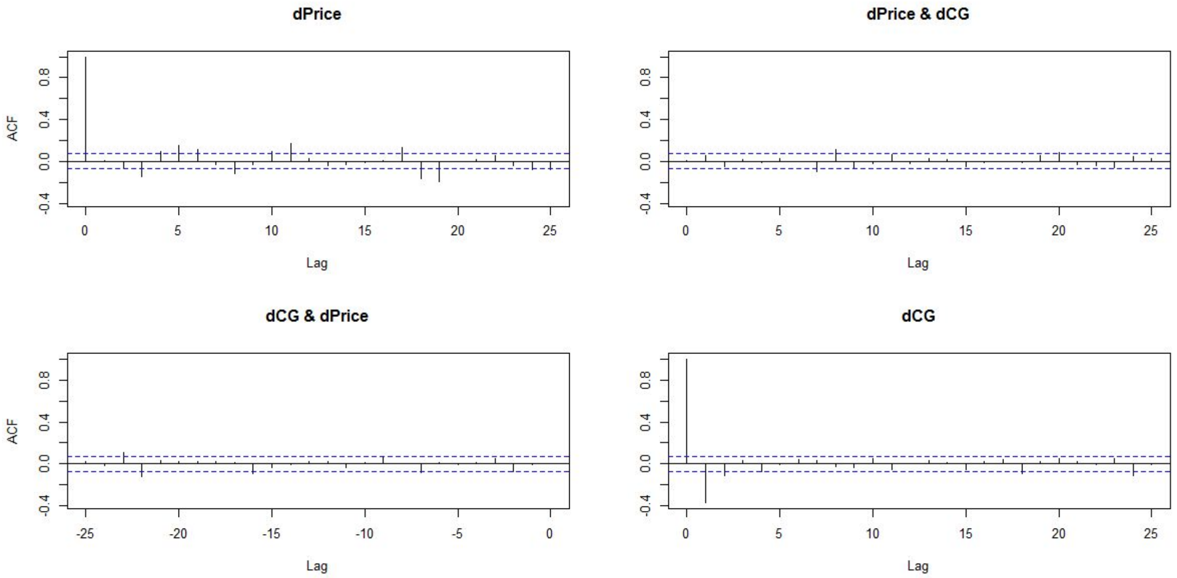
The following chart examines the temporal relationship between BTC price and the commit counts for CG Miner:



In the top left panel, we observe the CG Miner daily counts autocorrelation values at different lags. We see that there are statistically significant values at every lag examined. The bottom left panel shows the BTC price to be temporally correlated with itself. Neither of these insights contradicts common sense. In plain words, this means that the number of commits and the BTC price is correlated with the value on the previous date, and the value 2 days ago, etc. The top-right and bottom-left panels display the cross-correlation of CG Miner comments and the BTC Price at different lags. For example, for the top-right panel, we observe a correlation of 0.4 between the number of commits at time t and the BTC price at time t-1.

Because the latter analysis incorporates trends, we performed the same test on detrended data by analyzing the daily differences in number of commits and price.

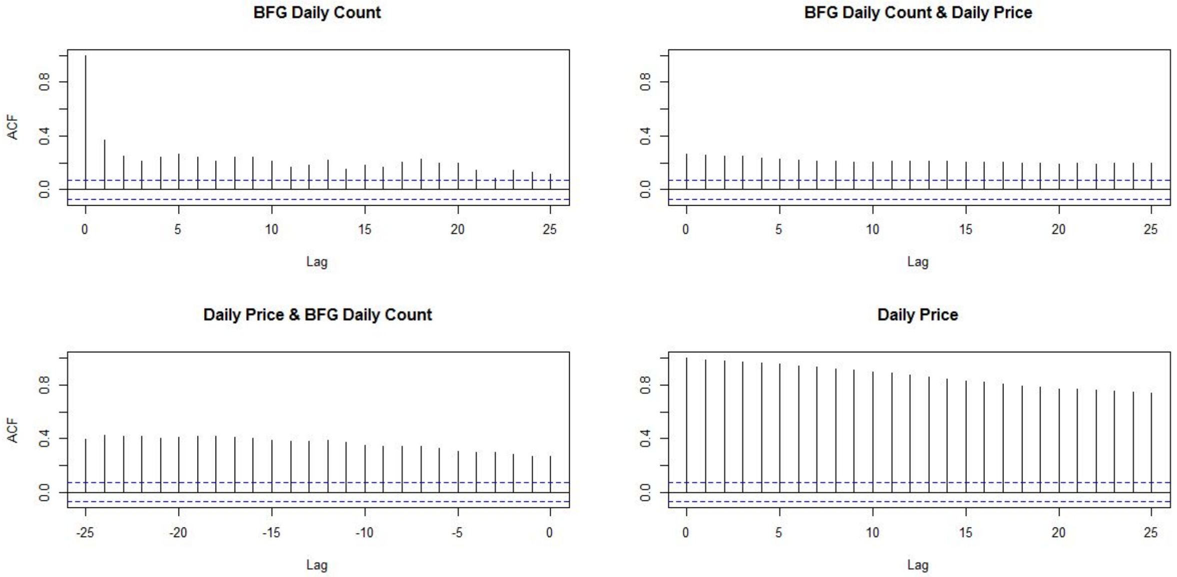
We observe an interesting trend in the 1st lag-correlation for CG Miner commits. The correlation is -0.4, which indicates that a positive difference in the number of commits is then followed by a negative difference. Thus, if today there are more commits than yesterday, then tomorrow there will be less commits than today.



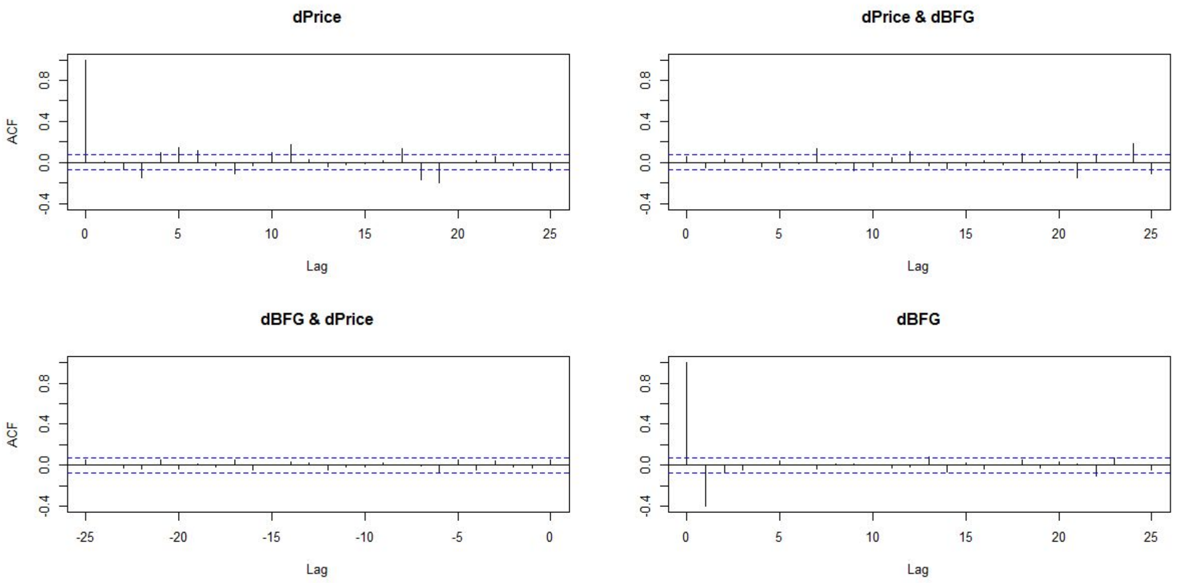
It is interesting to note that we do not observe any statistically significant cross-correlations between the detrended price and the detrended number of commits.

BFG Miner

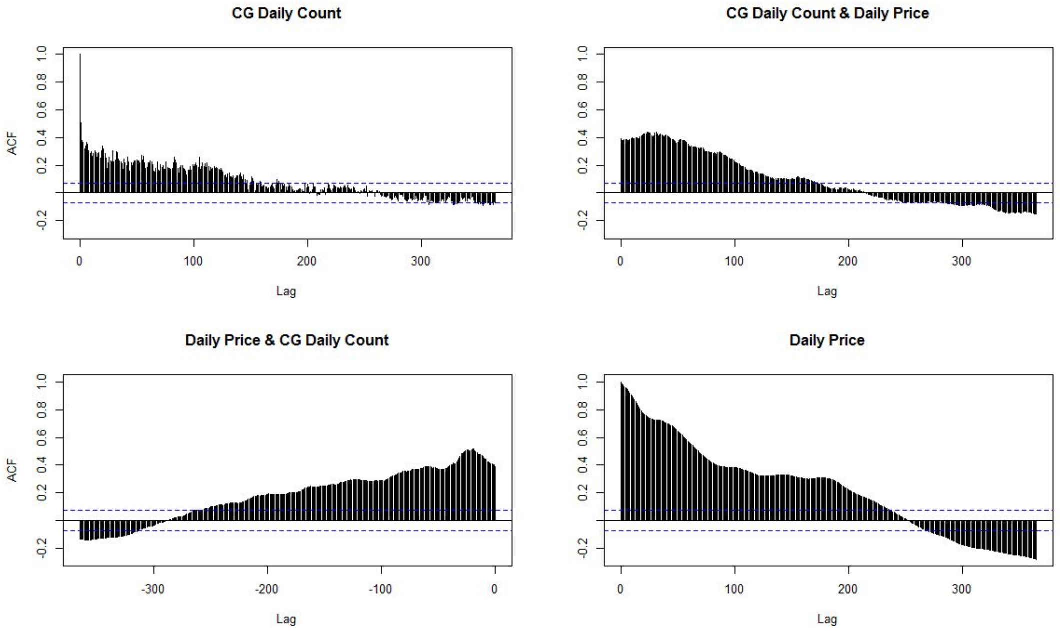
We observe very similar trends for BFG Miner, although the cross-correlation between daily counts and price is lower for BCG Miner than for CG Miner (0.3 versus 0.4):

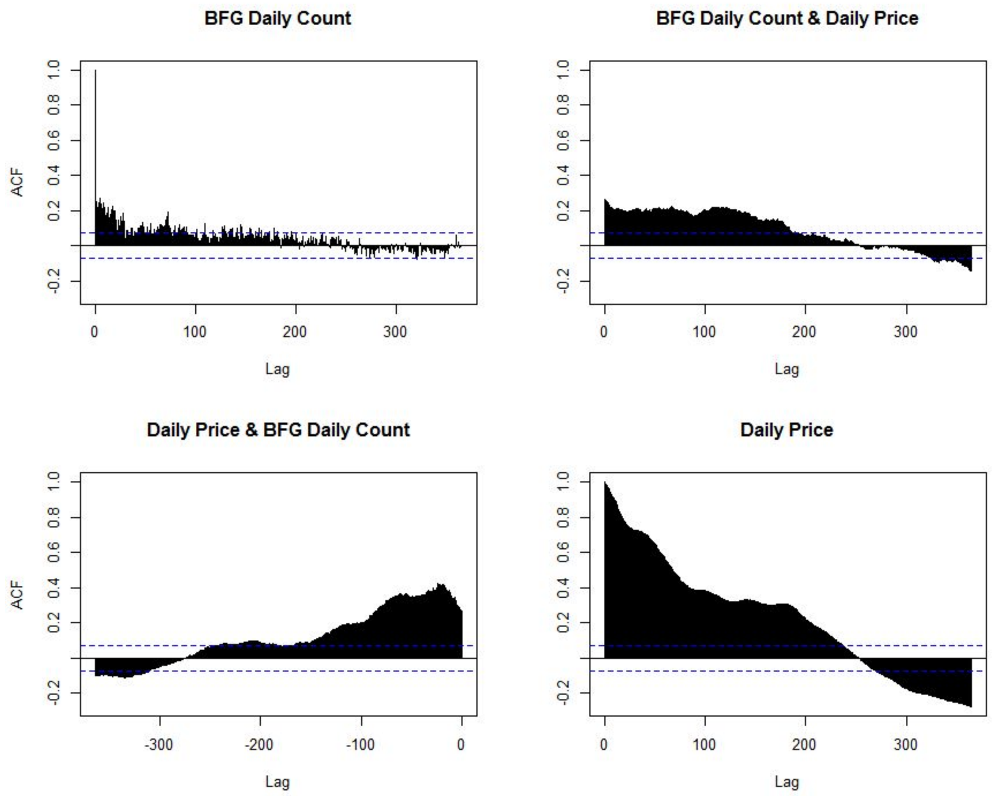


The detrended data comparing BFG with price exhibit the same phenomenon as the above analysis for CGMiner. Namely, we again see a negative autocorrelation at lag 1 for BFG Miner commits.



It is interesting to observe that the the autocorrelation become negative for both CGMiner , BFGMiner and price at lags over 200.





Granger Causality Test

A Granger-causality test runs an F-test between a y regressed on lagged y and lagged x and y regressed on itself only. If the F statistic is deemed to be statistically significant, x is said to Granger-cause y.

The results of the Granger Causality tests up to 30 lags indicate significant results for all 4 combinations:

```
                          F-statistic      p-value
price.Price -> cg.by.day.count    3.900105 5.044920e-11
cg.by.day.count -> price.Price    2.128798 4.962716e-04
```

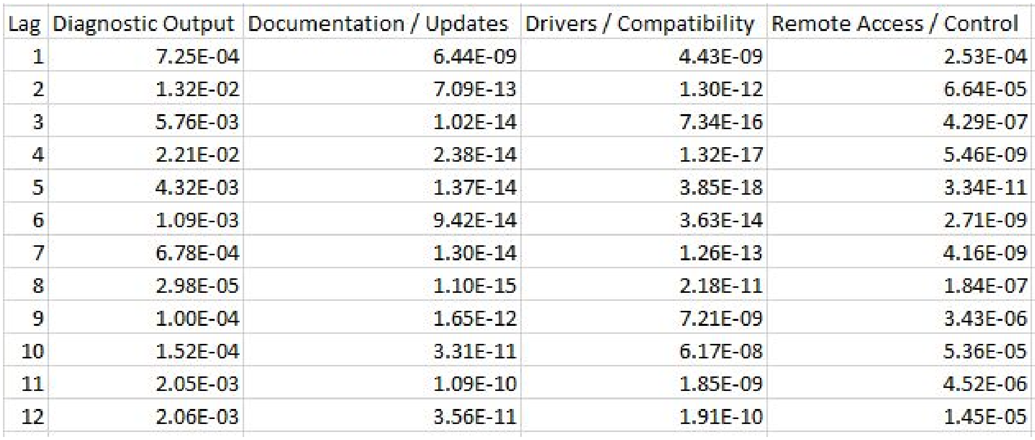
```
                           F-statistic      p-value
price.Price -> bfg.by.day.count    1.341392 1.073552e-01
bfg.by.day.count -> price.Price    2.483141 2.618836e-05
```

Modeling and Results

We regressed the number of commits in a particular category at each level of price lag up to 12 weeks for both CG Miner and BFG Miner. There were thus 96 models in total (2 miners X 4 categories X 12 lags).

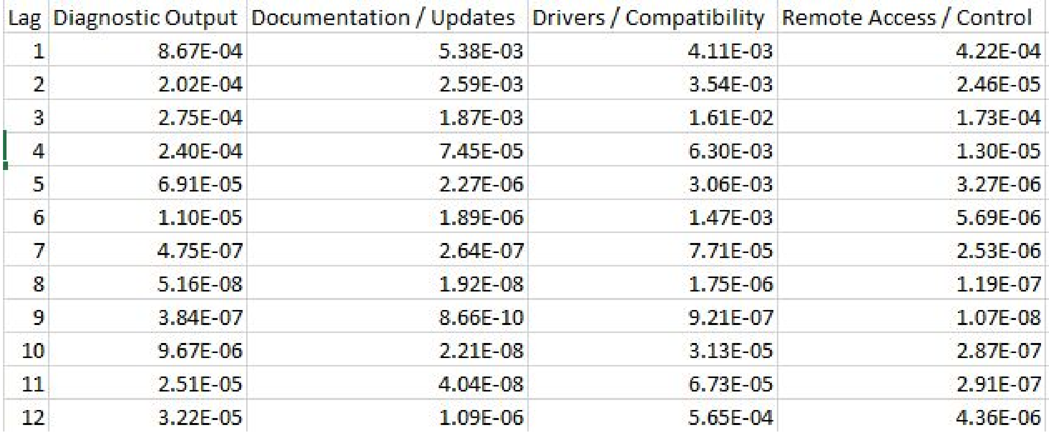
P-Values

The following table shows the p-values of the price coefficient at different lags for CG Miner:



Lag	Diagnostic Output	Documentation / Updates	Drivers / Compatibility	Remote Access / Control
1	7.25E-04	6.44E-09	4.43E-09	2.53E-04
2	1.32E-02	7.09E-13	1.30E-12	6.64E-05
3	5.76E-03	1.02E-14	7.34E-16	4.29E-07
4	2.21E-02	2.38E-14	1.32E-17	5.46E-09
5	4.32E-03	1.37E-14	3.85E-18	3.34E-11
6	1.09E-03	9.42E-14	3.63E-14	2.71E-09
7	6.78E-04	1.30E-14	1.26E-13	4.16E-09
8	2.98E-05	1.10E-15	2.18E-11	1.84E-07
9	1.00E-04	1.65E-12	7.21E-09	3.43E-06
10	1.52E-04	3.31E-11	6.17E-08	5.36E-05
11	2.05E-03	1.09E-10	1.85E-09	4.52E-06
12	2.06E-03	3.56E-11	1.91E-10	1.45E-05

We observe analogous statistically-significant p-values for BFG Miner:

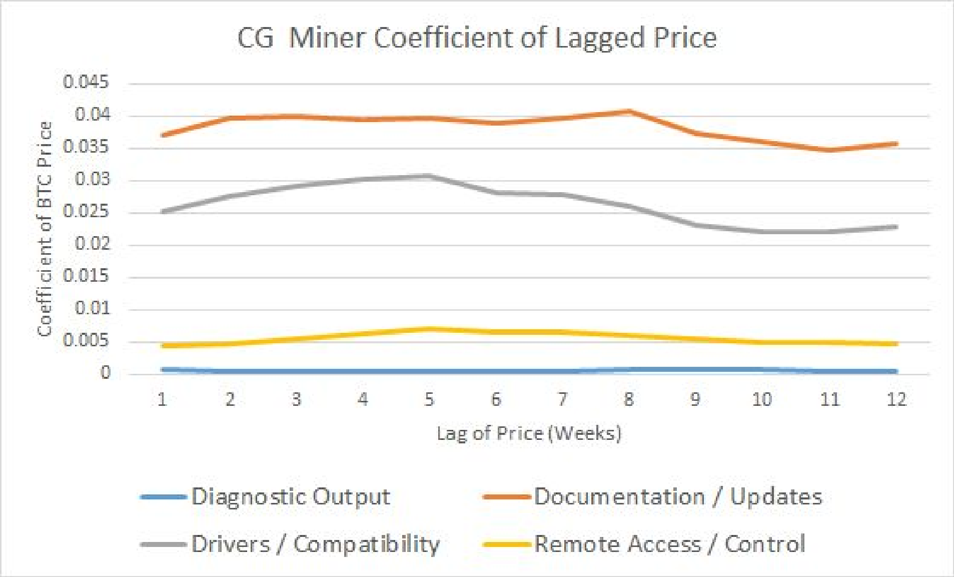


Lag	Diagnostic Output	Documentation / Updates	Drivers / Compatibility	Remote Access / Control
1	8.67E-04	5.38E-03	4.11E-03	4.22E-04
2	2.02E-04	2.59E-03	3.54E-03	2.46E-05
3	2.75E-04	1.87E-03	1.61E-02	1.73E-04
4	2.40E-04	7.45E-05	6.30E-03	1.30E-05
5	6.91E-05	2.27E-06	3.06E-03	3.27E-06
6	1.10E-05	1.89E-06	1.47E-03	5.69E-06
7	4.75E-07	2.64E-07	7.71E-05	2.53E-06
8	5.16E-08	1.92E-08	1.75E-06	1.19E-07
9	3.84E-07	8.66E-10	9.21E-07	1.07E-08
10	9.67E-06	2.21E-08	3.13E-05	2.87E-07
11	2.51E-05	4.04E-08	6.73E-05	2.91E-07
12	3.22E-05	1.09E-06	5.65E-04	4.36E-06

All of the p-values are observed to be statistically significant at all lags considered for all 4 categories of commits.

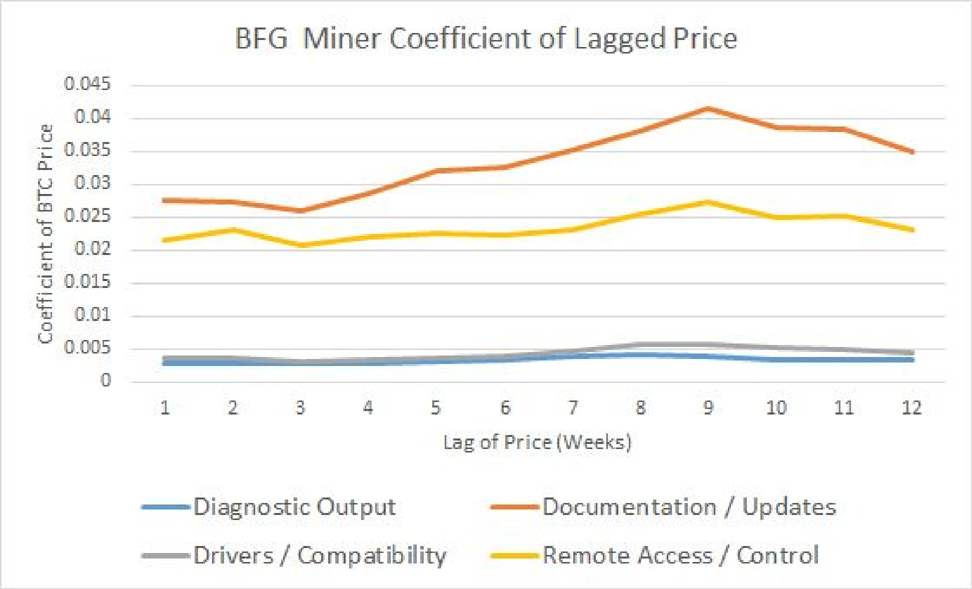
Coefficients

The table below analyzes the CG Miner coefficients of price at different lags for the 4 categories of commits:

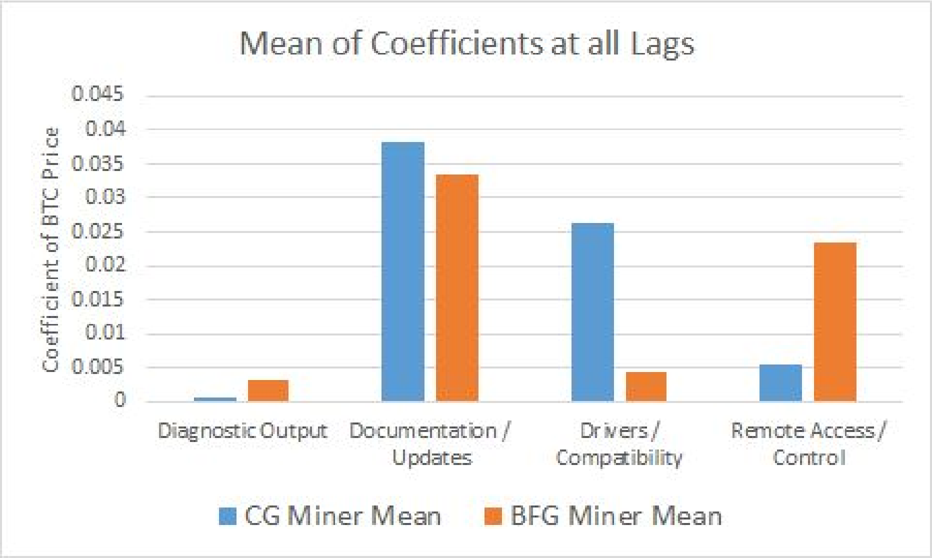


We observe that changes in BTC price have the greatest effect on changes in the number of Documentation / Updates and Drivers / Compatibility commits. It must be noted that the data for commits in the two remaining categories are sparse.

BFG Miner's coefficients are illustrated in the chart below:



The following chart compares the coefficients of each commit category between the two miners by averaging the coefficients from all lags.



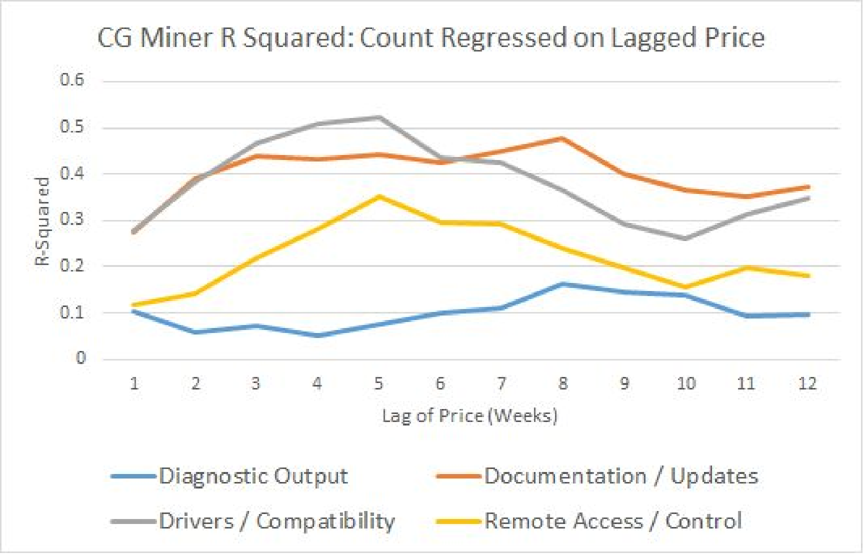
We observe consistency in the coefficients of the Documentation / Updates category among CG Miner and BFG Miner. The coefficients of the Drivers/Compatibility, Diagnostic Output and the Remote Access / Control categories, however, differ starkly between CG Miner and BFG Miner. Barring errors associated with proper categorization, this finding suggests a difference in the nature of changes to the miners.

For Documentation / Updates, the models with the highest Price coefficient, a $1 increase in BTC price results in an average increase of .4 commits for CG Miner 2 weeks later. Throughout the time period considered, the average week-to-week difference in the BTC price was approximately $40. We would expect to see a 1.6 increase in commits in the Documentation / Updates category associated with such a price fluctuation. On average, there are approximately 15 commits per week in the documentation / update category. The price effect thus represents an 11% increase in the number of weekly commits.

For BFG Miner, we observe a coefficient of 0.041 at lag 9. A 1.6 increase in commits associated with a $40 increase in the BTC Price would likewise constitute 11% of the 15.21 average commits for that category. Examining the effects on commits in the Remote Access / Control category, we see that a $40 increase in price yields a 1.09 commit increase 9 weeks later (40 X .02728). One commit constitutes 10% of the 10.75 mean weekly commits in that category.

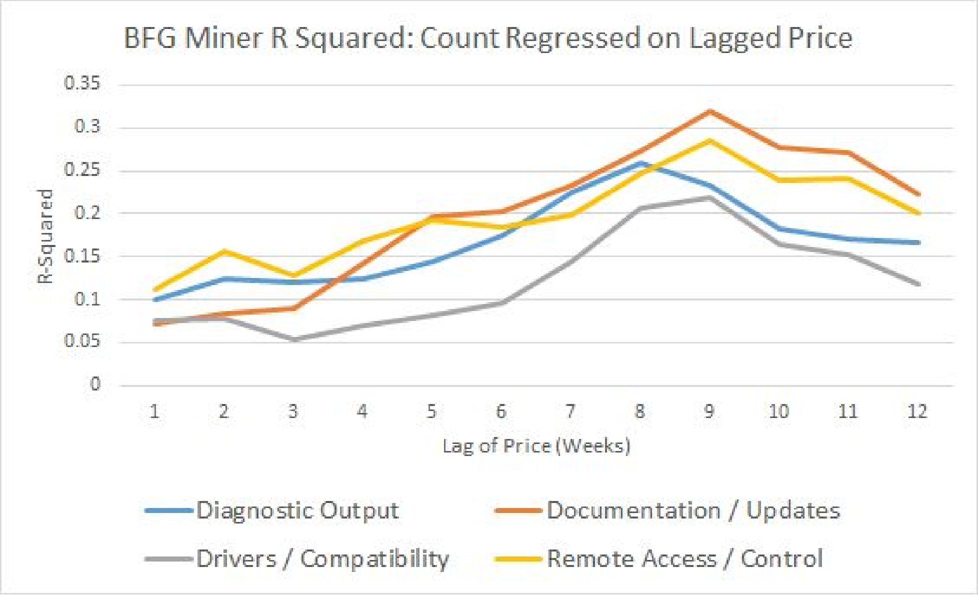
Explanatory Power

R-squared measure the percentage of variation in the response variable attributed to variation in the explanatory variable. The chart below shows the R-squared for the 12 lags and 4 response variables in the CG Miner Model:



We observe that at a lag of 5 weeks, changes in price explain 52% and 35% of the variation in the Drivers/Compatibility and Remote Access / Control categories. For the Documentation / Updates and Diagnostic Output categories, this maximum R-squared occurs at a lag of 8 weeks.

For BFG Miner, we observe that the R-squared is much lower for all categories. Also, the maximum R-squared occurs at a lag of 9 weeks for most of the categories.



An R-squared of 30-50% is generally considered high, especially since our models incorporate only one variable. The key insight from this analysis is the differing lags between CG Miner and BFG Miner. It appears that for the two miners, the maximum changes occur at different intervals following a price fluctuation.

Further Analysis

There were numerous items that could improve our analysis if implemented.

Of the 96 linear regression models, we noticed that the models on sparse elements like Diagnostic Output and Remote Access / Control violated the linear regression assumptions of normality and equal variance of errors. Although we did examine these assumptions for some of the more popular categories, it would be useful to conduct an investigation into which models adhere to the assumptions.

We trained our models using only 100 records out of approximately 2000 for CGMiner and 3000 for BFG Miner. Increasing the number of records in the training set would improve the accuracy of the classification model. Additionally, K Nearest Neighbor accuracy benefits from tuning the k parameter on a validation set.

Although the performance of the kNN algorithm is decent, aside from manually classifying the entire dataset, there is no way gauge its performance. Thus, it would be worthwhile to understand the causes for the unexpected output of the Naive Bayes system. We would be able to compare the result of the Naive Bayes classification to the kNN result to determine the accuracy.

Finally, it is common practice to include control variables in linear regression models. In our case, one obvious control variable would be the release of new mining hardware in a given week. Since BTC miners work hand-in-hand with hardware, new hardware releases often prompt BTC mining software updates.

Conclusion

In our initial investigation of the relationship between BTC price and changes to bitcoin mining software, we have determined 4 major categories of updates: diagnostic output, documentation / updates, drivers / compatibility, remote access / control. We then focused on two of the most popular BTC miners: CG Miner and BFG Miner.

In our analysis, we identified that 30-50% of the variation in the number of commits in a given category of is attributed to a change in price. The coefficients of the models at different lags and categories were all statistically significant, and an average weekly change in the BTC price results in a 10% increase in the number of commits for non-sparse categories like Documentation / Updates.

We also saw that while the highest R-squared for CG Miner categories occurs approximately 5 weeks after the price change, BFG Miner comments respond greatest 9 week following a price change.

Future work on BTC miners, aside from improving this analysis, would involve mining statistics for the number of people actually using particular mining software and understanding the features of proprietary bitcoin miners..