Université Jean Monnet
Faculté des Sciences et Techniques
Master Informatique
Parcours Machine Learning and Data Mining
Cohort 2022-2024

Data Mining for Big Data
Project

Data Mining for Big Data : Project

Students :
Hedi Zéghidi
Mohamed Moudjahed
Erick Gomez
Milan Jankovic

Professor :
Baptiste Jeudy
Christine Largeron

Contents

1 Introduction

The Bitcoin market in the period under study presents a positive trend and a number of market players in constant growth to later on concentrate the transactions in fewer hands. The information on actor behaviour each day is synthesized in new features that reflect distribution properties (max, min, average, median, standard deviation, skeweness, and kurtosis). We apply different models to predict the USD Bitcoin price: Random forest, robust linear regression with stepwise selection, ARIMA, LSTM and a baseline model that predicts the price as equal to the price the day before. The community detection task evaluates the communities created by the algorithms Louvain, Greedy, and Label Propagation. The dynamic nature of the communities requires the creation of different networks for different time periods.

1.1 Overview of the dataset

1.1.1 Prediction Task

In the Fig.1, we observe the price in USD of bitcoin during the year 2015

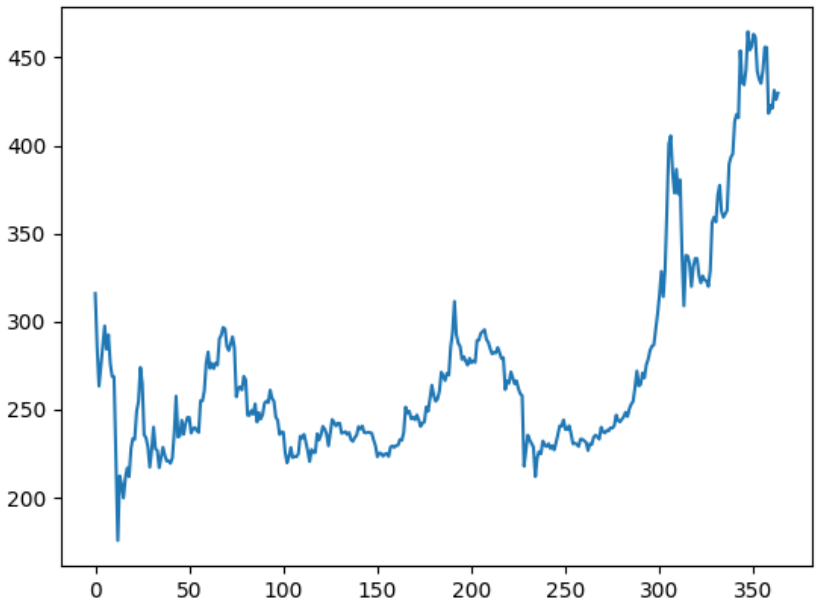
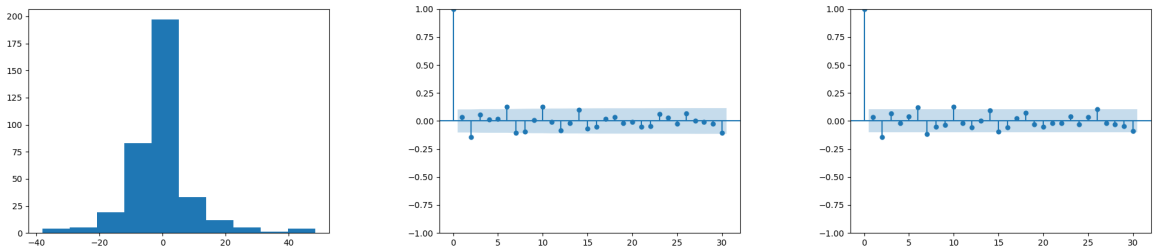


Figure 1: PriceUSD during the year 2015

In Fig.2 (a), we observe a normal distribution between the price and its lagged value. In (b) we plot the autocorrelation of this difference and in (c) the partial autocorrelation. Both diagrams suggest white noise, as is expected in price series.



(a) Price change per day (b) Auto correlation of (a) (c) Partial Auto corr. of (a)

Figure 2: Analysis of correlations of daily price changes

The prediction task uses a dataset composed of three data frames, all of them contain the year 2015 and they have in in common 5 columns: week (an integer identifying a week), weekday (integer from 0 to 6), year, month and day.

The main dataset is external.csv which contains only two columns, they do not come from the Bitcoin blockchain, but concern the Bitcoin economy:

- PriceUSD: the average observed value of a Bitcoin in US dollars over the day on the main exchange platforms and the value which will predicted
- HashRate: a measure of the total computing power used by miners

The second one is blockchain_global.csv with 14 columns and are aggregated data, calculated from Bitcoin blockchain data:

- nb_transactions: Number of transactions
- nb_payments: Total number of payments. A transaction can correspond to several payments.
- total_received_satoshi: Sum of values received
- total_sent_satoshi: Sum of values sent
- total_fee: Sum of amounts paid in transaction fees
- mean_fee_satoshi: Average fees paid per transaction
- mean_feeUSD: Average fees paid, in USD
- mean_fee_for100: Average transaction amounts paid in fees
- mean_nb_inputs: Average number of exits per transaction
- mean_nb_outputs: Average number of entries per transaction
- nb_mining: Number of mining
- total_mining_satoshi: Total amounts collected by miners (newly created BTC and transaction fees)
- newly_created_coins: Number of newly created coins received by miners

- self_spent_satoshi: Total sums that the actors return to themselves

And the final one are blockchain_by_actor.csv with the 100 actors with the greatest activity (defined in number of days of activity) in the Blockchain community of this period:

- identity: Actor ID, which can be a unique name or number
- received: Total amounts received
- spent : Total amounts paid
- nb_received: Number of transaction outputs received by the actor
- nb_transactions: Number of transactions made by the actor
- nb_spent: Number of payments made by the actor (1 transaction = 1 or more payments).
- sum_fee: Total transaction fees paid by the actor for transactions for which he is the source
- mean_fee_for100: Average fees paid per transaction
- self_spent: Amounts observed as sent from the actor to himself
- self_spent_estimated: Amounts estimated as probable sent from the actor to himself, but to addresses that we do not know.

1.1.2 Analysis of the Community Structure of the Network

Contrary to the first task, we have 2 years and 6 months from 2015/01/01 to 2017/06/30 of data in the second task. Each day is represented by a csv file which represent directed networks, in the form of a list of links. Each line corresponds to a link, and represents a summary of the exchanges between 2 actors during the day. It is composed of the following columns:

- Source: The source actor sending the money
- Target: The actor who receives the money
- Value: Sum of the amounts sent by the Source actor to the Target actor during this day (in satoshi).
- nb_transactions: The number of transactions made by Source to Target

2 Task 1: Prediction Task

2.1 Data preparation

2.1.1 Data grouping

1. We have loaded the three files "external.csv", "by_actor.csv" and "global.csv"

2. We removed unnecessary columns in DataFrames. For example in global we removed the columns "xxxx", "year", "day", "month", "original order".

3. We merged the "external" and "global" DataFrames using the common "week" and "weekday" columns to create "external_global".

4. We performed data aggregation on the "actor" DataFrame by grouping by "week" and "weekday". Then, we calculated the mean, standard deviation, minimum and maximum for each group.

5. We merged the "external_global" DataFrame with the "actor_grouped" DataFrame on the "week" and "weekday" columns to get "external_global_actor".

6. Finally, we removed the "week" and "weekday" columns from the final "external_global_actor" DataFrame.

2.1.2 Feature Creation

The variables "by actor" present the market behaviour of the 100 biggest agents in terms of transaction value. We capture this behaviour by constructing features that represent distribution characteristics of each of the columns in the "by actor" file. For each column we create the minimum, maximum, average, median, skewness, kurtosis, and standard deviation. We merge this new features to the data where we have the Bitcoin price for each day.

Another source of features are the lagged values of the variables, including the price. Since we do not have access to the week order in the test data for which we have to deliver predictions, we take lags no longer than six. Another feaure we create is the division between hashrate and the newly created coins to give a proxy for the cost of production. Finally, to summarize the distribution of number of transactions in "by actor" file, we compute the normalized distribution and the Wasserstein distance between consecutive days

2.1.3 Preparing training and testing data

To predict the price of bitcoin, we gathered 6 days of data and used the 7th day's price as a label. We created two datasets, X and y.

1. Set X contains information about the first 6 days of each period.

2. The set y contains the price of bitcoin for the 7th day of each period.

We separated our data into two groups: 80% for training and 20% for testing. This division was carried out while preserving the chronological order of the data, to avoid any form of "data leakage". By respecting this temporal separation, we ensured that the data used to train the model does not contain information from the future, which could have misled our model by giving it access to knowledge that it should not. to have.

2.1.4 Features selection

Columns containing exclusively zero values have been eliminated. These are the columns where the minimum values were consolidated during the aggregation of the "actor" table. Additionally, columns with a 100% correlation between them were also removed, as this means that there is a factor connecting these two columns, which would be useless for training our model. However, we have decided not to remove highly correlated columns (positively or negatively) unless the correlation reaches 100%; removing two highly correlated columns may result in the loss of important information, as each of these columns may provide a unique perspective or specific details in the data.

Stepwise regression approach

Feature selection is essential in linear regression because it simplifies the model, avoids overfitting, improves the understanding of influential factors, reduces data redundancy, and increases the reliability of predictions. We have decided to use a stepwise approach, which is an iterative method for selecting the most important variables in a multiple linear regression model. In our case, we are based on the coefficient of determination R^2 (R-squared), which measures how well the model explains the variation in the dependent variable (target variable):

$$R^2 = \frac{SSR}{SST}$$

SSR : Sum of Squares of Regression, measuring the variation explained by the model.
SST : Total Sum of Squares, representing the total variation in the dependent variable.

1. **Forward Selection:** We start with an empty model (intercept only) and iteratively add variables that significantly improve the R^2 until no significant improvement can be obtained.
2. **Backward Selection:** After including all potentially relevant variables, we can remove those that do not significantly contribute to improving the R^2 until the model can no longer be simplified.

The goal is to obtain a multiple linear regression model that is simpler and more explanatory by retaining only the variables that truly contribute to explaining the variance in the target variable (high R^2) or by minimizing another specific performance criterion. This helps to avoid overfitting and build a more robust model.

Using Stepwise Regression

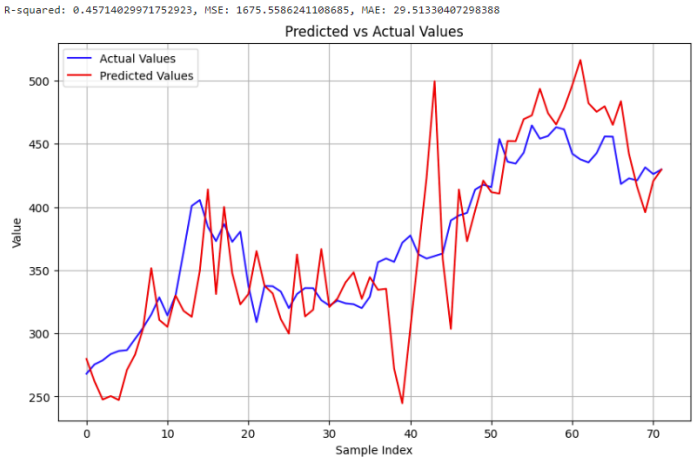


Figure 3: Comparing predictions with test data without using stepwise regression

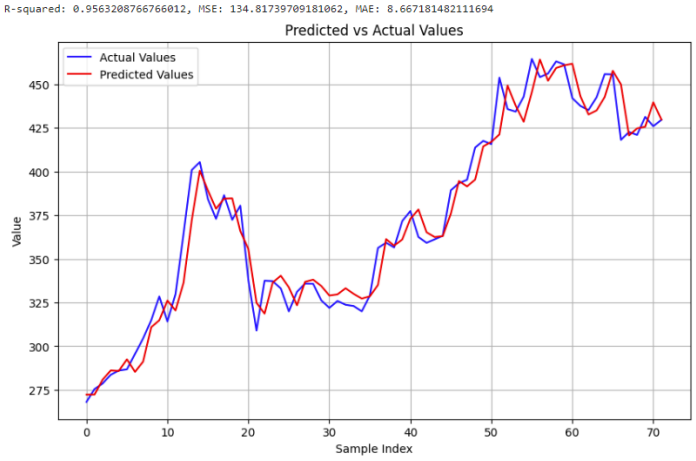


Figure 4: Comparing predictions with test data using stepwise regression

With 324 features for 6 days, our initial model had an R^2 score of 46%. However, thanks to a careful selection of 46 features, our R^2 score rose to 96%. This shows that appropriate feature selection can greatly improve model performance by reducing noise and focusing on the most influential factors.

2.1.5 Random Forest Feature Selector

A different approach we had for feature selection was to use random forests. Instead of just being used as a prediction model, we used its training to then extract the models' feature importances, to then be used as a feature selector.

Our approach was to train a Random Forest model as a regressor using the 2015 datasets we had at hand. Once the training was finished, we had a look at the feature importance of said model. The Random Forest model gives a feature importance score which reflects the contribution of each feature for its predictions. This can be used as an intuitive metric for finding features that contribute significantly to the predictions of the model.

However, while the model could work very well for predicting on one dataset, its results were woeful when training on one dataset and testing on another with negative r2 scores. In other words, when the model is trained on the 2015 dataset with the dataset cut in two (80% training, 20% testing) the model gets very good results at 0.93 r2 score. Same goes when it is trained on the task1 2016 dataset, where it produced similarly good results. However, when the model is trained with the 2015 dataset, and tested on the 2016 dataset, it simply does not do well at all. This is the reason why, in the end, we did not choose the random forest model as a feature selector.

One interesting thing to note is that when comparing the feature importances after training on each of the two respective datasets, they vary quite a lot, which explains

why the model struggles to make any good predictions. For the 2015 dataset by far the most important feature was "PriceUSD lag 1", the price of the previous day. On the other hand, when trained on the 2016 data, the with a similar amount of importance, the most important feature by far was div_HashRate_newly_created_coins, a column we created, which represented the division between the hashrate and the amount of newly created coins.

2.2 Description of the prediction models

1. **Baseline Model :** Our Baseline approach only consists of taking the price of bitcoin from the previous day. This follows human logic, people who invest in cryptocurrencies tend to primarily look at the price of the most recent course.

2. **ARIMA Model :** We used an ARIMA (AutoRegressive Integrated Moving Average) approach to forecast Bitcoin prices. ARIMA is a temporal forecasting model that takes into account trends and seasons in the data. It consists of three main components: autoregression (AR), integration (I) and moving average (MA). Autoregression captures linear relationships between past values, integration helps make the data stationary, while moving average deals with sequential dependencies in the data. By automatically adjusting the parameters of these components to the training data, ARIMA is able to generate Bitcoin price predictions.

3. **Robust linear regression Model :** Linear regression as used in 2.1.4 is a statistical technique used to model the relationship between a dependent variable and one or more independent variables, by fitting a linear equation to the data. This method predicts the value of the dependent variable from the values of the independent variables, assuming a linear relationship between them. To improve the robustness of this technique against outliers and atypical data, we can add a robust linear regression standard, such as:

    - **HuberT:** Mitigates the impact of extreme values by combining the squared error for small errors with the absolute error for large ones, providing a good compromise between sensitivity to outliers and statistical efficiency.
    - **RamsayE:** Adjusts the influence of observations based on their deviation from prediction, with a focus on reducing the impact of very outlying values, allowing flexible adaptation to extreme data.
    - **AndrewWave:** Uses a sine function to limit the impact of outliers, gradually reducing their weight beyond a certain threshold, which helps handle extreme data while remaining sensitive to moderate data.
    - **TrimmedMean:** Excludes a certain percentage of the most extreme values on both sides of the data distribution before calculating the regression, which helps minimize the effect of outliers on the analysis.
    - **Hampel:** Identifies outliers based on interquartile range and adjusts their influence accordingly, providing a robust method that is particularly effective against widely dispersed anomalies.

For our experiments, we decided to use the HuberT robust linear regression standard. This choice was based on the most satisfactory results obtained with this standard during preliminary tests.

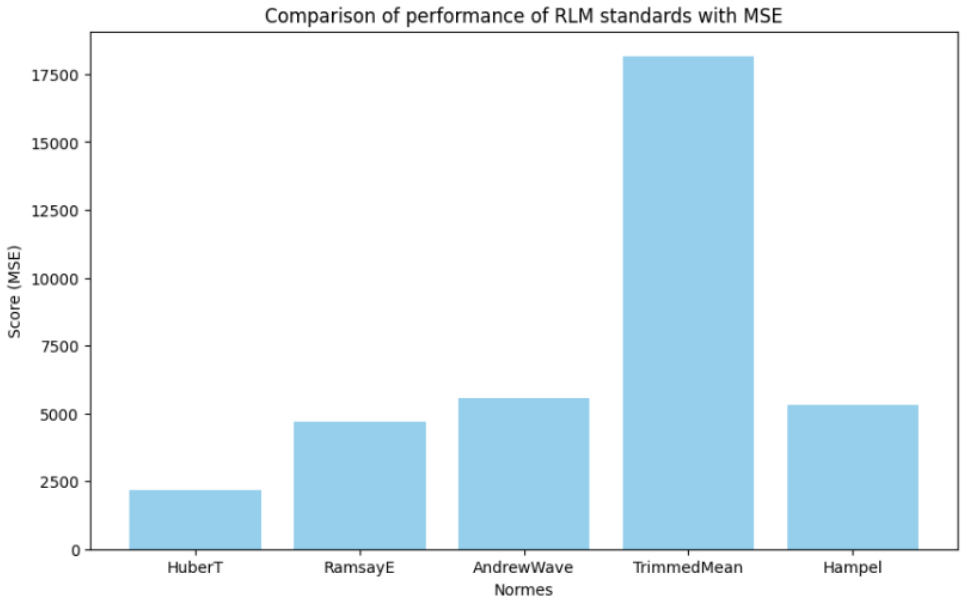


Figure 5: Comparison of performance of RLM standards with MSE

4. **LSTM Model** We also used an LSTM model to predict the price of Bitcoin, exploiting its ability to capture long-term dependencies in time series data. Our simple architecture includes:

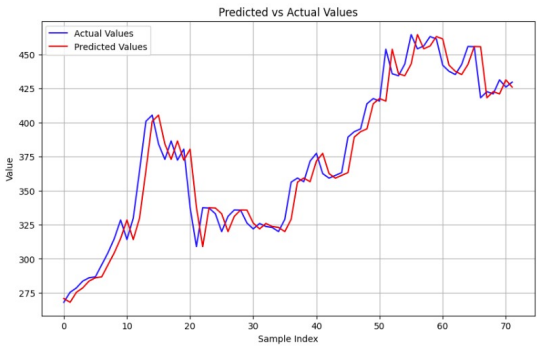
   - **An LSTM layer** for learning data sequences, with an 8 unit.
   - **A Dropout layer** at 0.05 to reduce overfitting by randomly eliminating connections.
   - **A Dense layer** that produces the final prediction.

   The model is compiled with the 'Adam' optimizer with a learning rate of 0.001 and uses MSE as the loss function.

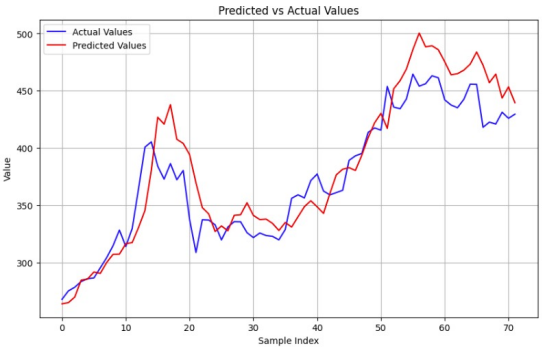
2.3 Model performance in 2015 data

	MSE	MAE	R2-score
Baseline	217	10	92%
ARIMA	282	12	91%
Stepwise Linear Regression	134	8	95%
Robust Linear Regression	79	5.6	97%
LSTM	640	20	79%

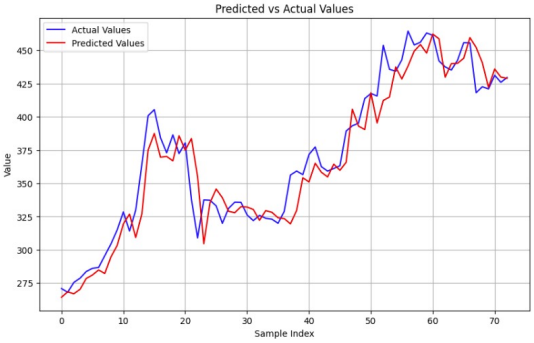
Table 1: Mean Squared Error (MSE), Mean Absolute Value (MAE), R2-score,



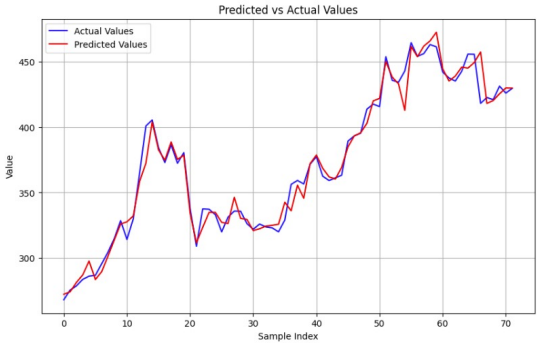
(a) Baseline model



(b) LSTM model



(c) Arima



(d) Robust Linear Regression

Figure 6: Predicted vs Actual value with the different model

To predict the price of Bitcoin, we tested different models. The first, very simple, just used the previous day's price. It worked well, with a good R^2 score of **92%**.

Then, we tried a linear regression, it did better than the simple model, with an R^2 score of **95%**.

But it is robust linear regression, which is more resistant to abnormal data, which gave the best results, with a very good R^2 score of **97%**.

The ARIMA and LSTM models, often used for this type of prediction, were not as efficient in our case. ARIMA had a slightly worse R^2 score than the simple model, and LSTM was the least effective with an R^2 score of **79%**.

As in the Fig.7, we have the validation and training loss for the LSTM model where we observe that the two losses declines sharply until it reaches 0. For the evaluation, we use the model with lowest validation loss.

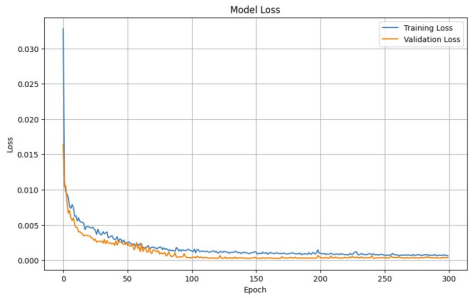


Figure 7: LSTM loss for the training and validation

3 Task 2: Community Detection

The data reflect the market operation for each day from January 1st of 2015 to June 30th of 2017. A node is an agent that either buys or sellls Bitcoin. The direction of the transaction gives birth to and edge. Therefore the graph is directed. For a better representation of the community dynamic we divide the time frame under study in 30-day length periods. We prefere to take into account smaller time periods to determine whether there are changes in the network, and if those changes are related to the movements of the Bitcoin price. In Fig 8 we present the number of active nodes in the market for each 30-day period. By active node we mean a node that made a sell or bought bitcoin. A smaller number of nodes could point to a smaller number of market transactions, however, in Fig 9, the number of transactions does not follow the drop in the number of nodes we observe in Fig 8. From Figures 9 and 8 we can conclude that a reduction of market players goes together with an increase in the number of transactions for each player.

In Fig. 10 we observe a drop at the middle of the period, when the Bitcoin price suffers a price decrease after a strike of constant growth. As we mention above, the number of transactions keeps stable as the number of players decreases. However, we can also conclude that the transactions are for smaller Bitcoin denominations around August 2016, and they later start to increase again.

After this brief description of the Bitcoin market, we inquire in the following application of community detection about market concentration and its relationship to the price movements. In Fig. 11, we observe that the degree distribution is skewed to the right, meaning that a majority of nodes have a low degree (less than five in this case). The networks across all 30-day length time periods are sparse and consist mostly of nodes with few connections.

We apply three algorithms for community detection: Louvain, Label Propagation, and Greedy. The Louvain algorithm Blondel et al. [2008] is a modularity-based approach that optimizes the modularity of a network, which measures the density of links within communities compared to links between communities. It optimizes iteratively moving nodes between communities to maximize modularity locally. We choose to apply the Louvain algorithm because it is known for its efficiency and scalability. We also consider that modularity is a good measure of the frequence of transactions between the same market players (nodes).

The Label Propagation Raghavan et al. [2007] algorithm is a simple and fast algorithm that relies on the idea that nodes with the same label (initially assigned) are likely to belong to the same community. The labels are assigned according the majority of the labels of the neighbors and continues iteratively until convergence. Certain network structures, such as the presence of highly connected nodes (hubs) or the existence of bridges between communities, can impact the performance of Label Propagation. In the following pages we will see how centrals nodes arise and become hubs. This behaviour impacts negatively the performance of Label Propagations.

Greedy Algorithm (Girvan-Newman Algorithm) Girvan and Newman [2002] works by iteratively removing edges with the highest betweenness centrality, which measures how

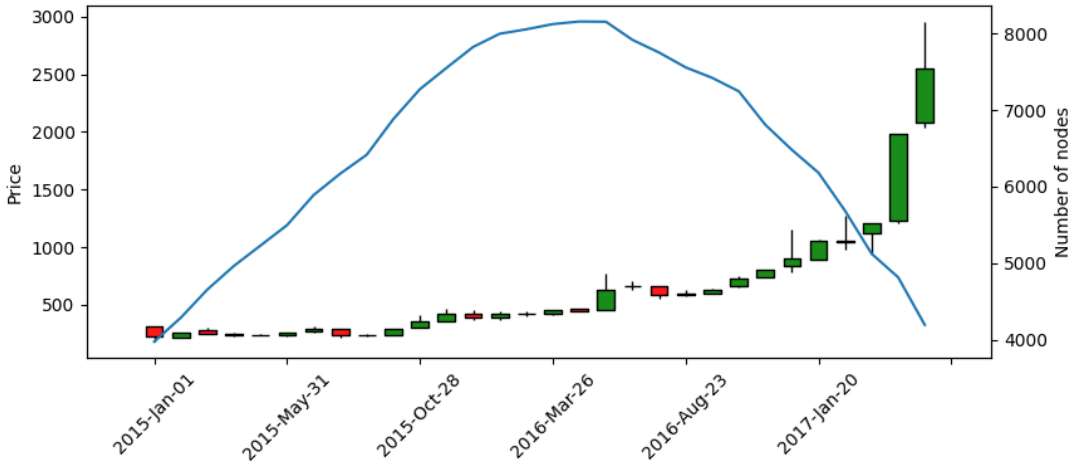


Figure 8: The price is in candlestick. The color of the rectangle is red (green) when the price at the end of the period is lower (higher) than at the beginning. The number of nodes is the blue line, with values on the secondary y axis. The number of nodes in each 30-day length period increases until the middle of the frame time under study. We observe a drop in the number of nodes when the price stops increasing in June-July of 2016. The reduction in the number of nodes can point to market concentration.

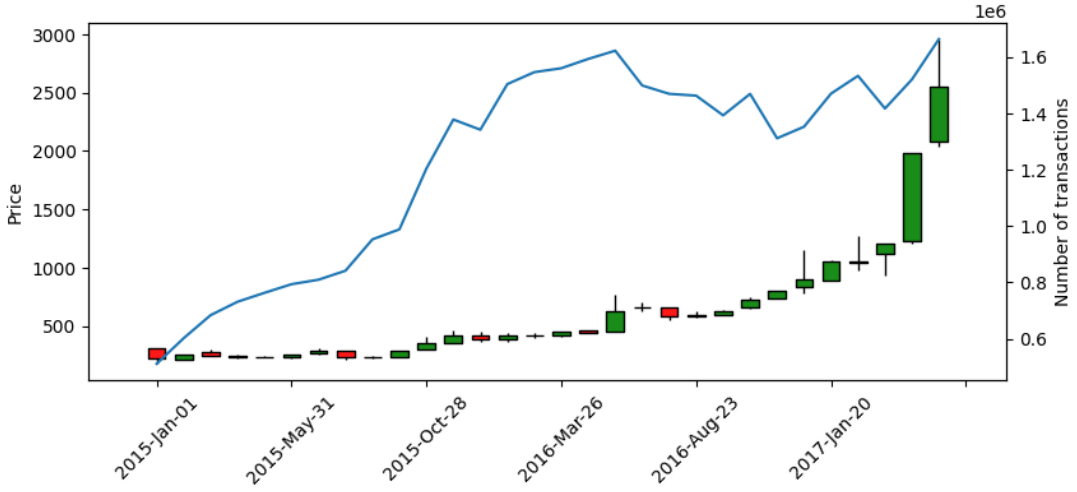


Figure 9: The price is in candlestick. The color of the rectangle is red (green) when the price at the end of the period is lower (higher) than at the beginning. The number of transactions is the blue line, with values on the secondary y axis. We observe an increasing number of transactions, that follows a slight drop correlated with the price drop on June-July of 2016. Afterwards, the number of transactions stabilizes.

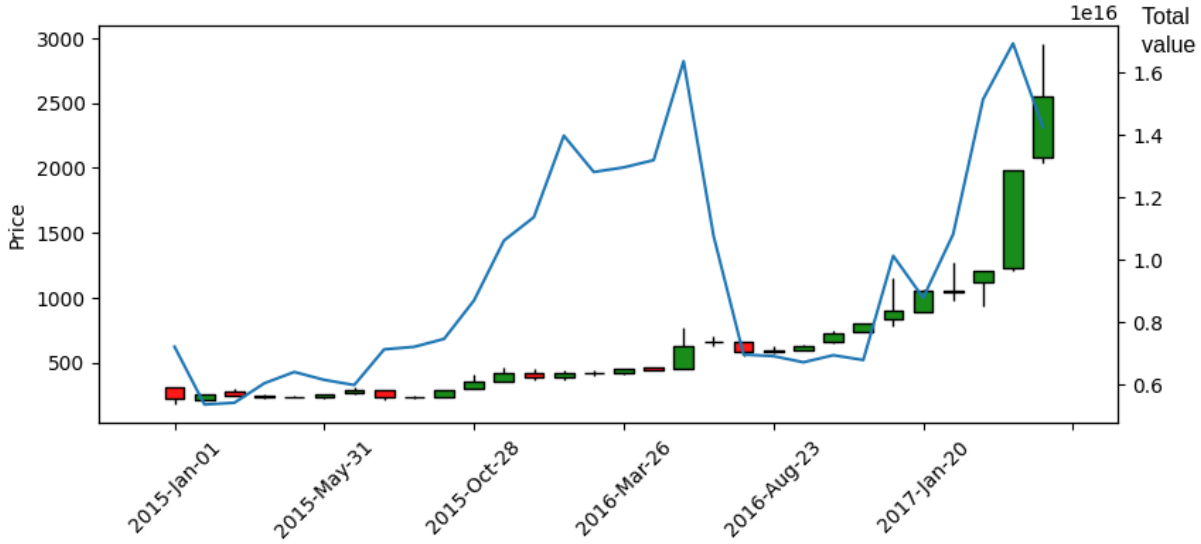


Figure 10: The price is in candlestick. The color of the rectangle is red (green) when the price at the end of the period is lower (higher) than at the beginning. The total transacted value is the blue line, with values on the secondary y axis. We observe a linear trend on the first half, and an acute drop that coincides with the sudden increase and fall of the Bitcoin price. In the second half, the total exchanged value increases mirroring the price explosion.

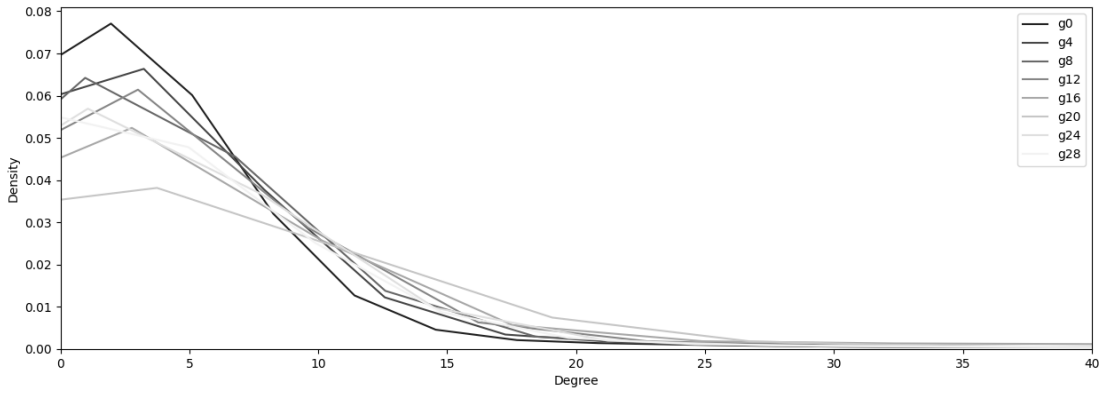
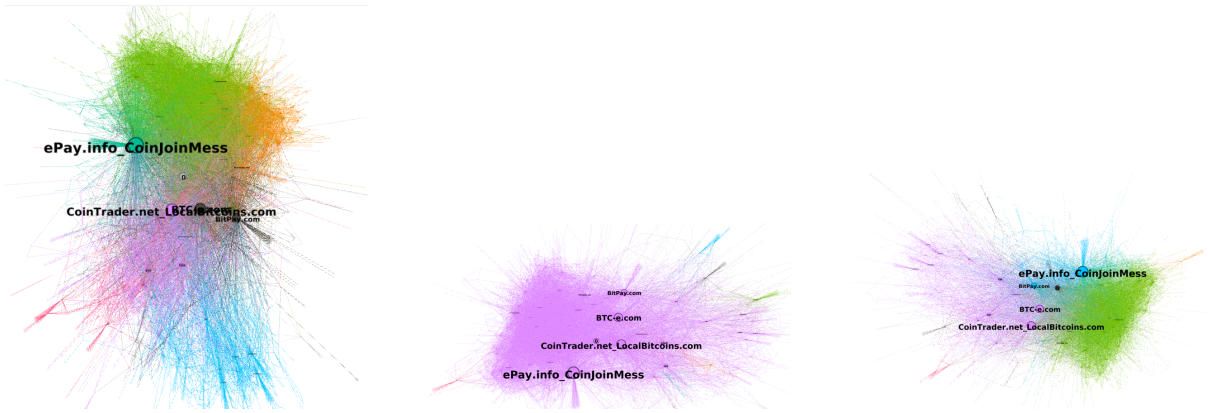


Figure 11: The degree distribution is less concentrated as the time passes, until the middle of the period we study, where increases again. Here g1, g4, etc, correspond to the 30-day length periods.



(a) Louvain (b) Label Propagation (c) Greedy

Figure 12: Community detection for the first 30-day period (2015-01-01 to 2015-01-30). Label size of the node in proportion to betweenness centrality.

often an edge lies on the shortest path between pairs of nodes. Unlike the Louvain algorithm, the Greedy approach can be computationally expensive.

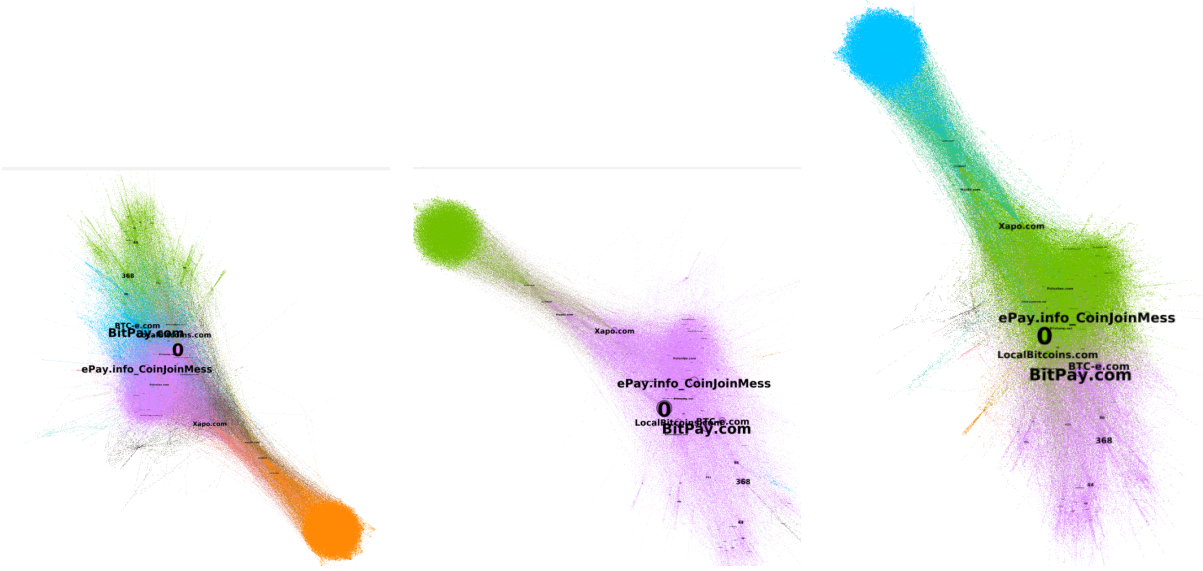
In Fig. 12 the three chosen algorithms detect communities for the first 30-day length period (from 2015-01-01 to 2015-01-30). The size of the nodel label is proportional to betweenness centrality. The Label Propagation algorithm fails do detect communities of different sizes and concentrates the majority of the nodes in one community. Greedy presents three commnunites and Louvain five. In both cases, there is an absence of hubs for each community.

After the consecutive strike of Bitcoin price growth from the end of the year 2015 to the middle of 216, specifically on 2016-04-25, the communities we detect by the three different algorithms agree on the presence of a low density community, without a main hub. The node "*ePay.info_ CoinjoinMess*" is constantly a major player. The node labeled "0" could represent transitory actors without a particular label. Since we do not have information on the nature of the label "*0*", we do not remove it from the analysis.

If we focus ourselves on the communities after the market correction, i.e on 2016-04-25, we find that Label Propagation once again fails to detect different communities, due to a drop in modularity (see Fig 20). The drop in price cools down the market and a reduced number of players have lower density.

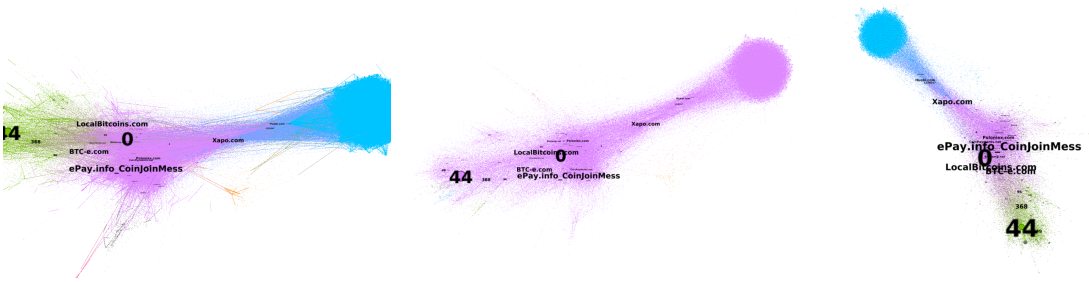
Fig. 15 presents the in-degree distribution for each 30-day length network. During the first half of the total period, the distribution shows a median equivalent to the first quartile. Half of the nodes only buy Bitcoin once during each 30-day length period. While the Bitcoin price increased consecutively, the number of transactions where a node buys bitcoin climbs to two. This behaviour is later on reverted, and a quarter of the nodes do not buy bitcoin.

The out degree distribution (see Fig 16) shows more spread that the in degree distribution (see Fig. 15) At the height of the market increase, 75% of the nodes sell Bitcoin



(a) Louvain (b) Label Propagation (c) Greedy

Figure 13: Community detection for the 17^{th} 30-day period (2016-04-25 to 2016-05-24). Label size of the node in proportion to betweenness centrality.



(a) Louvain (b) Label Propagation (c) Greedy

Figure 14: Community detection for the 19^{th} 30-day period (2016-07-24 to 2016-08-22). Label size of the node in proportion to betweenness centrality.

at least four times. This preference for selling reflect that a majority of the market agents prefer to pocket their earnings while the hubs start to accumulate Bitcoin wealth.

The median degree is lower than the average degree 17, showing a characteristic of Scale-free networks where only a small proportion of nodes have a great number of connections. The average degree drops precipitously after the market price correction in the months following July 2016. In short, for the most part, half of nodes buy only once, while half of the nodes sell at least in two opportunities (median out-degree equal to two while median in-degree equal to one). More sellers than buyers and a few hubs mean Bitcoin accumulation in a few hands.

In Fig. 18 we present the entropy of the distribution of community size by number of nodes in each community for the communities we detect using different detection algorithms. Lower entropy points to less communities, but with an increasing number of nodes, proportionally. The market concentration stops in June of 2016 and almost recovers the initial level it has at the beginning of 2015. The entropy of the communities we detect with Label propagation is more variable than the Louvain or Greedy entropies.

Regarding the relation between the log of community size measured by the number of nodes and the mean degree in each community, we observe a linear relationship, more patent for the communities we detect using the Louvain algorithm (see Fig 20). If we assume that the networks present a scale-free structure, where a few nodes have significantly higher degrees than others, the logarithmic relationship could be a characteristic feature. In scale-free networks, the size of communities may increase logarithmically with their mean degree. Also, the linear relationship might imply that communities with higher mean degrees are denser and have more internal connections. This could be associated with networks where nodes within bigger communities are more tightly connected.

We now inquire about the difference or similarity between the communities we find in a network with respect to another network. We compare networks that come from consecutive time periods, and from the different community detection algorithms. We expect that the partitions we find in a network are similar to the the partion from the period before. We also expect to relate the difference with the price behaviour. We calculate the difference between the partition of communities using the Jaccard, Normalized Mutual Information (NMI), and Adjusted Rand Score. We observe for all three measures, although with different volatility, an increasing dissimilarity between communities, followed by a decreasing dissimilarity at the end of the constant price increment in June 2016. The higer volatility of the Jaccard score can be explained by changes in the community sizes and low sparsity. Besides, and in contrast to the NMI, the Jaccard score does not normalize for the size of the communities.

Fig 22 presents the NMI between communites found using the Louvain algorithm. It compares networks that are consecutive in time. In the y-axis we present the NMI index we obtain when we compare the community partition of a network using as source a 30-day length period and another network that uses as source the preceding 30-day length period. In the x-axis we present the Bitcoin price in USD. The blue dots correspond to the networks constructed with data coming from dates before July 2016. The red dots

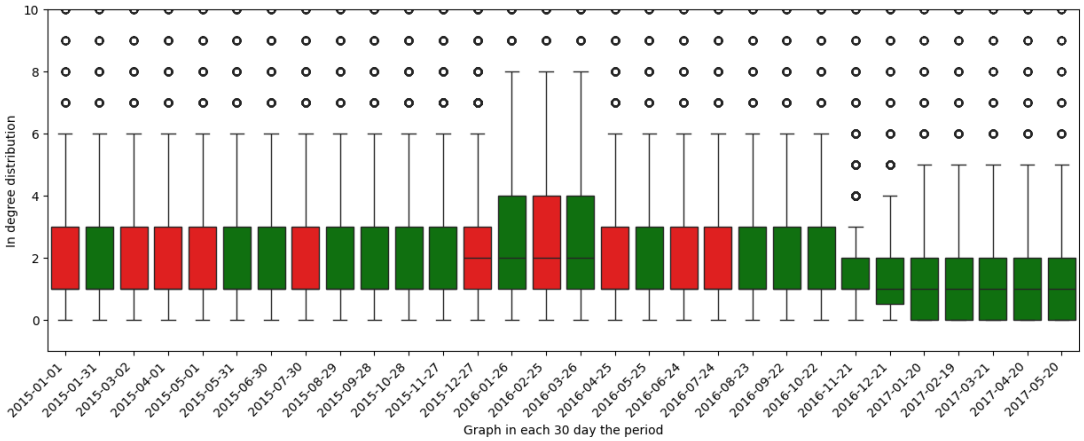


Figure 15: The in-degree distribution for the first half of the period we study shows a median equivalent to the first quartile. Half of the nodes only buy Bitcoin once during each 30-day length period. During the long term increase of bitcoin price, the number of transactions where a node buys bitcoin jumps to two. This behaviour is later on reverted, and a quarter of the nodes do not buy bitcoin.

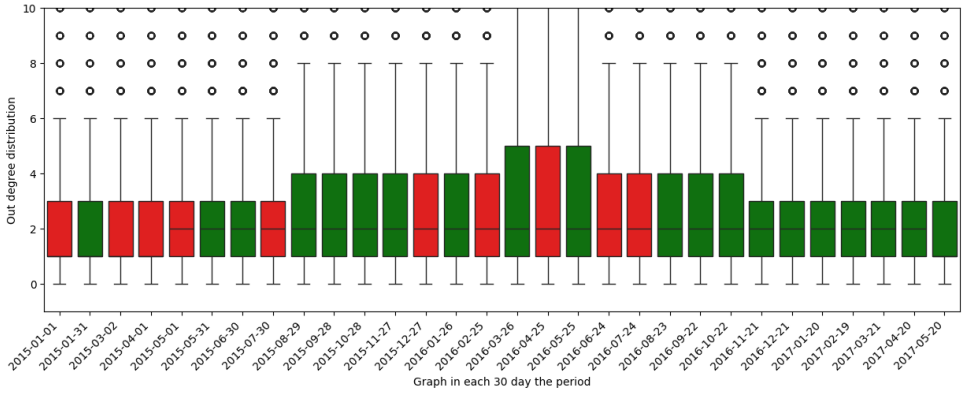


Figure 16: The out degree distribution shows more spread that the in degree distribution. At the height of the market increase, 75% of the nodes sell Bitcoin at least four times.

are from networks using dates after July 2016. We observe a positive and linear relationship between differences in the communities we find using the Louvain algorithm and the Bitcoin price until July 2016.

In Figures 12, 13, and 14, the size of the label is proportional to the betweenness centrality. Among the many possible centraty measures, we choose betweenness without discriminating for direction because the data reflects market transactions and a hub could be buying and selling for a margin, without regard for the identification of the seller or buyer.

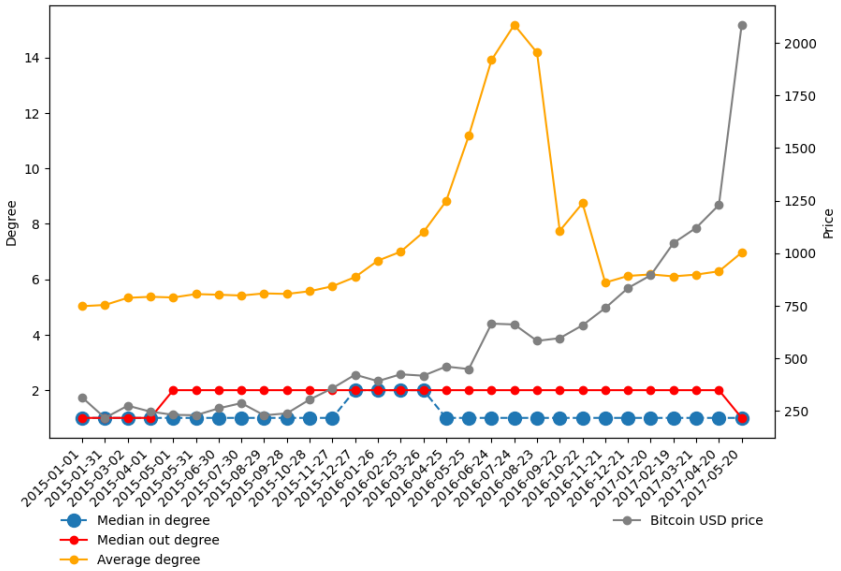


Figure 17: The median degree is lower than the average degree, showing a characteristic of Scale-free networks where only a small proportion of nodes have a great number of connections. The average degree drops precipitously after the market price correction in the months following July 2016. In short, for the most part, half of nodes buy only once, while half of the nodes sell at least in two opportunities (median out-degree equal to two while median in-degree equal to one)

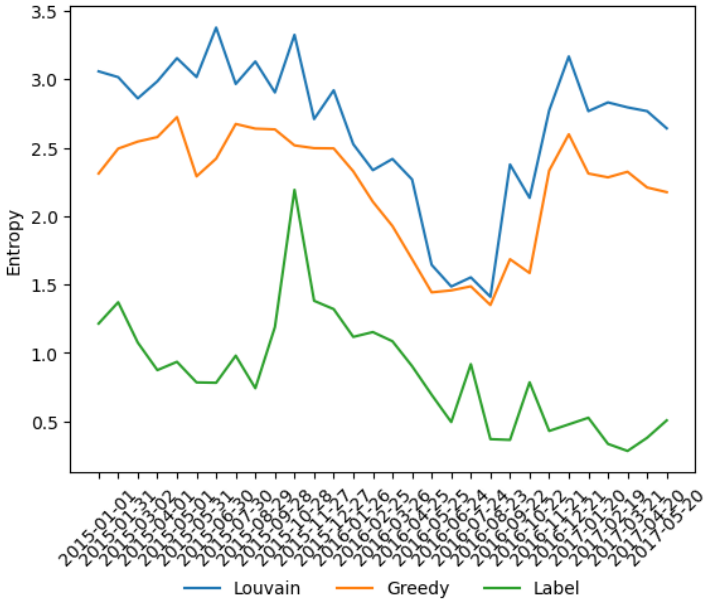


Figure 18: Entropy of the community size distribution for different community detection algorithms. Lower entropy points to less communities, but with an increasing number of nodes, proportionally. The market concentration stops in June of 2016 and almost recovers the initial level it has at the beginning of 2015. The entropy of the communities we detect with Label propagation is more variable than the entropy from Louvain or Greedy communities.

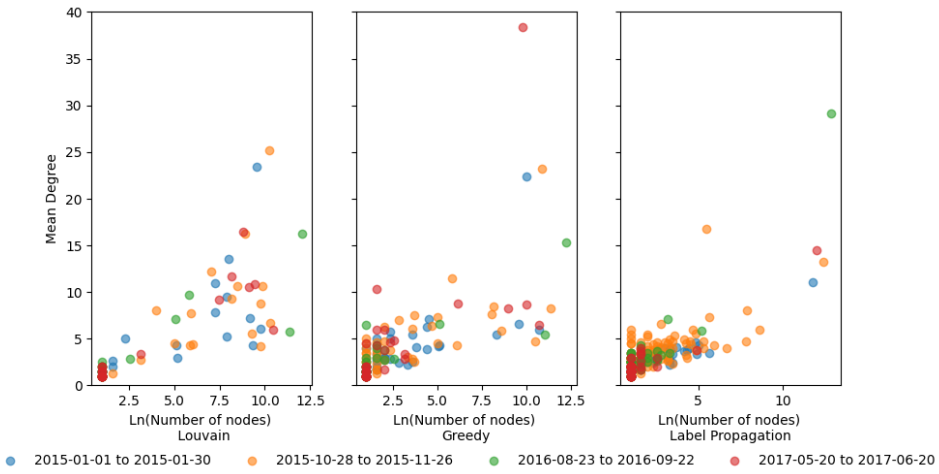


Figure 19: The log of community size in number of nodes and the mean degree in each community for different periods. The relationship we observe using the values provided by the Louvain algorithm exhibits a linear relationship between the natural logarithm of the size of communities and the mean degree in a network. If we assume that networks present a scale-free structure, where a few nodes have significantly higher degrees than others, the logarithmic relationship could be a characteristic feature. In scale-free networks, the size of communities may increase logarithmically with their mean degree. Also, the linear relationship might imply that communities with higher mean degrees are denser and have more internal connections. This could be associated with networks where nodes within bigger communities are more tightly connected.

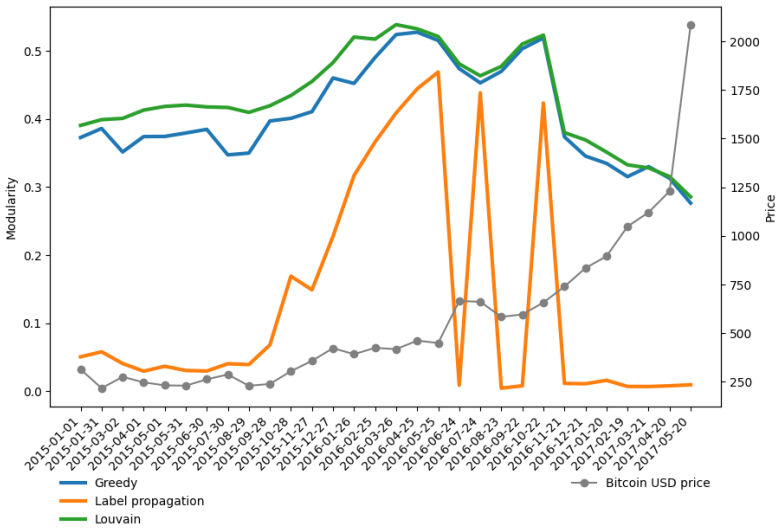


Figure 20: The modularity of each of the 30-day length network for the communities we detect using Greedy, Louvain, or Label Propagation.

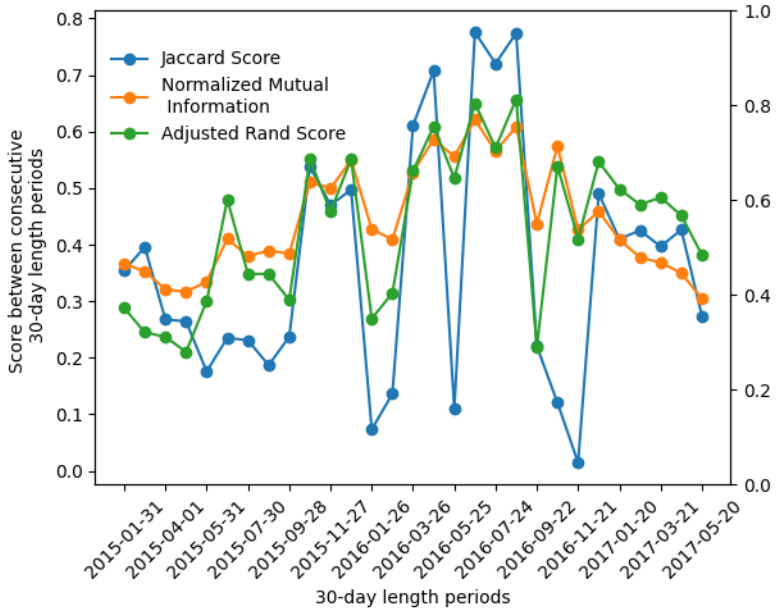


Figure 21: The communities we find in each network from an specific 30-day length period are different than the communities we find from the preceding 30-day length period. We calculate the difference between the partition of communities using the Jaccard, Normalized Mutual Information, and Adjusted Rand Score. We observe for all three measures, although with different volatility, an increasing dissimilarity between communities, followed by a decreasing dissimilarity at the end of the constant price increment in June 2016.

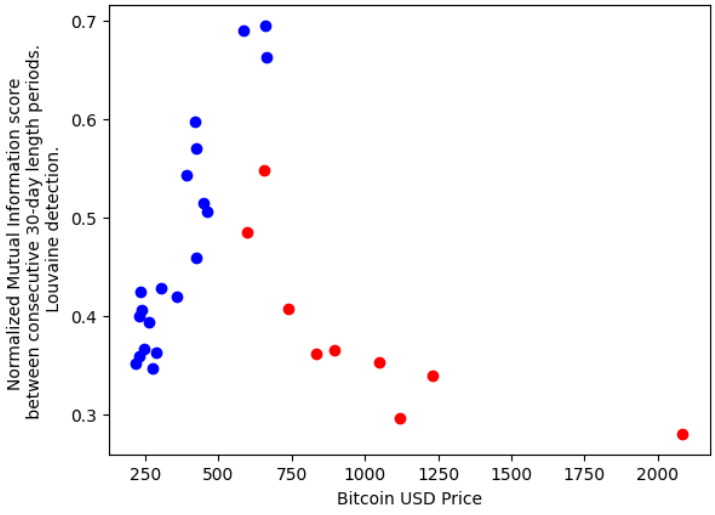


Figure 22: In the y-axis we present the NMI index we obtain when we compare the community partition of a network using as source a 30-day length period and another network that uses as source the preceding 30-day length period. In the x-axis we present the Bitcoin price in USD. The blue dots correspond to the networks constructed with data coming from dates before July 2016. The red dots are from networks using dates after July 2016. We observe a positive and linear relationship between differences in the communities we find using the Louvain algorithm and the Bitcoin price until July 2016.

3.1 Conclusion

The analysis covers market data from January 1st, 2015, to June 30th, 2017, divided into 30-day length periods for a better representation of community dynamics. The market is represented as a directed graph, where nodes represent agents buying or selling Bitcoin, and edges represent transactions. We construct a network for each 30-day length period. For the detection of communities we remove the direction. The previous analysis leads as to the following conclusions

- **Market Dynamics:** A reduction in the number of market players is observed to be accompanied by an increase in the number of transactions for each player. A drop in the total value of Bitcoin transactions is observed around August 2016, coinciding with a market correction. However, the number of transactions remains stable.
- **Community Detection Algorithms:** Three community detection algorithms —Louvain, Label Propagation, and Greedy—are applied to analyze the network's community structure. The Label Propagation, in comparison, fails to detect communities when the modularity decreases.
- **Characteristics of Detected Communities:** The degree distribution is observed to be skewed to the right, suggesting that a majority of nodes have a low degree. Scale-free network characteristics are noted, with a small proportion of nodes having a high number of connections (hubs). Analysis of in-degree and out-degree distributions reveals changing patterns in buying and selling behaviors over time.
- **Entropy and Community Structure:** The entropy of the distribution of community sizes indicates changes in market concentration, with a decrease and subsequent recovery observed.
- **Linear Relationship between Community Size and Mean Degree:** A linear relationship is observed between the natural logarithm of community size (number of nodes) and the mean degree, particularly pronounced when using the Louvain algorithm. This suggests a potential scale-free structure.
- **Comparison of Community Partitions:** The Jaccard score, Normalized Mutual Information (NMI), and Adjusted Rand Score are used to compare community partitions over consecutive time periods. Increasing dissimilarity between communities is noted, followed by a decreasing dissimilarity after a period of constant price increment in June 2016.
- **Relationship Between Community Differences and Bitcoin Price:** The NMI between community partitions found using the Louvain algorithm shows a positive and linear relationship with differences in Bitcoin prices until July 2016.

In summary, we provide insights into the dynamic nature of the Bitcoin market, the behavior of market participants, and the evolving community structure. The use of various network metrics and community detection algorithms contributes to a comprehensive understanding of the market's temporal and structural characteristics. Additionally, the exploration of the relationship between community dynamics and Bitcoin price variations offers valuable insights into the market's response to external factors.

References

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.