2020-06-14

Using Group Affinity to Predict Community Formation in Social Networks

Joseph Leung

Honors Thesis

USING GROUP AFFINITY TO PREDICT COMMUNITY FORMATION IN SOCIAL NETWORKS

by
Joseph Leung

Submitted to Brigham Young University in partial fulfillment of graduation requirements for University Honors Department of Mathematics

Brigham Young University
May 2020

Advisor: Ben Webb
Honors Coordinator: Michael Griffin

Abstract

USING GROUP AFFINITY TO PREDICT COMMUNITY FORMATION IN SOCIAL NETWORKS

Joseph Leung

Department of Mathematics

Bachelor of Science

A well-studied topic in network theory is detecting the communities found in real-world networks. Community detection is a technique to better understand the way in which small dense substructures appear in these networks. Such substructures can often tell important information about groups that form in such systems. A prominent feature of many networks is that they evolve over time, forming and dissolving new edges between different nodes that appear. In this thesis, we consider how we can use the community structure of a network at a certain point in time to predict the state of a network's communities at some time in the future. Through the use of "affinity scores" that describe a node's inclination to be part of a community, we predict the formation of future communities in the network with the assistance of machine learning algorithms. Using the method proposed in this thesis, we find that it is possible to predict, with a moderate degree of accuracy, the communities that will eventually form in a network before they are fully formed.

Contents

List of Figures

1 Introduction

1.1 Basics of Network Science

Network theory is a branch of mathematics that has gained increased popularity, partly due to the rise of social networks in recent years and many other day-to-day applications ranging from internet usage and the World Wide Web to family trees. A network is represented by a graph $G = (V, E)$ with vertices (or nodes) V and edges E. Nodes represent the individual elements or actors of a network, and edges represent the relations or interactions between the nodes. For example, in a social network individual persons act as the nodes in a network. They either follow or are friended by other individuals, as represented by an edge in the network. Though a simple representation, it is easy for these structures to become quite complicated, as seen in Figure 1 which represents the Southern Women Data Set [1].

Though networks may seemingly come from different sources, it is not uncommon to find that these different networks can have surprisingly similar forms. For example, when considering biology and technology, one might be surprised that a network representing how technology progresses have a similar structure to a network that represents biological evolution [2]. These kinds of surprising connections are what many researchers have made efforts to understand.

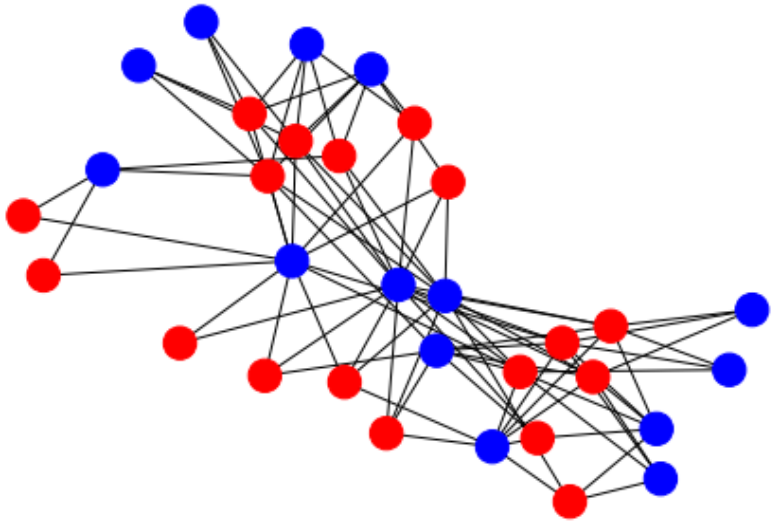


Figure 1: Southern Women Data Set

An example of a relatively simple network consisting of 18 women and 16 events they attended in the course of a year in what is referred to as "Old City." Red dots represent women and blue dots represent the events they attended

1.2 Evolving Structure of Networks

Many networks evolve over a period of time, with new nodes and edges appearing or disappearing in different parts of the network. With social networks, relationships are continuously forming and dissolving, with new persons establishing relationships which are constantly appearing or leaving. As a result, understanding how these network structures change is an important and not well-understood part of network science, and creating predictive models of these changes is part of how The United States National Research Council defines network science [3].

There are many attempts to discern how these networks evolve, one of which is through the problem of link prediction, which is the idea of trying to predict where

edges may eventually form within a network [4]. A common example of this is on social networks such as social media sites, where one may see recommendation of who to follow or friend. As one might imagine, this ability to predict is often quite dependent on the complexity of the network itself and how much information we have to describe the network. For example, the social network LinkedIn uses many factors when recommending someone to connect with, such as if two individuals work at the same company or graduated from the same university.

In this thesis, we explore further how we can predict the state of an evolving network. Rather than just predict links between individual nodes, we focus on how we can predict group formation within a network, also known as the network's community structure.

2 Community Detection Methods

An important method to better understand networks is through their community structure, that is whether nodes in a network can be gathered into their own "communities" which may or may not overlap. In this thesis, we focus on the case of non-overlapping communities, which implies that a network naturally divides into distinct communities. Understanding a network's community structure is important because it allows us to understand the way the network functions which facilitates the study of the network, and explore further characteristics of the network itself. A salient example that motivates the study of community structure is that of modeling

the spread of disease. Knowing which subsets of a population in which a disease may be especially potent helps health professionals act effectively to contain the rate of infection without infringing on the liberties of other groups that are less affected. In many cases, understanding community structure can also improve the efficiency of link prediction as mentioned in Subsection 1.2, as it can simplify what different nodes may or may not be connected.

An integral part of understanding community structure is the process of detecting communities. Community detection is the method by which we try to partition a network into different groups, and it is worth noting that much research does exist regarding this topic. There are many different community detection methods which are computationally expensive or use brute force methods, many of which can be found in several literature reviews [5]. Most techniques focus on a network's modularity, detecting densely connected substructures within a network. Intuitively, these techniques often remove edges (sometimes randomly, sometimes not) to see what clusters of nodes, i.e. communities, remain connected [6]. Other methods utilize the neighbors of a node (the nodes closest to a particular node) to try and see what neighborhoods are most likely to exist [7]. Other methods use the idea of centrality to see which nodes may be most integral to a network, and then base communities off of these measures [8]. Needless to say, many techniques exist, and their effectiveness along with their corresponding computational complexity often depends on many factors such as the size of a network or the structure itself. [5].

In this thesis, we utilize four different community detection methods for vary-

ing purposes: Kernighan-Lin Bisection, Spectral Partitioning, K-Clique Neighbors, and Fluid communities. We describe properties of these algorithms in detail in the following subsections. How we used them to determine affinity scores is discussed in Section 5.

2.1 Kernigahn-Lin Bisection

The Kernighan-Lin Bisection (KL) algorithm is a stochastic modularity based algorithm that essentially cuts a network in two many times until it finds one cut that best isolates two different communities [9], which also implies that this method only discovers two communities. Every iteration of this algorithm begins with a random bisection of the network, thus it is very possible that different iterations result in different outcomes. However, it can be costly to run this algorithm repeatedly, as it has a temporal complexity of $O(n^2\log(n)$, where n is the number of nodes in the network.

2.2 Spectral Partitioning

This method, which also only finds two communities, takes what is called the Graph Laplacian (see Definition 3.1 and Figure 2) of a network, and uses the eigenvector corresponding to the second smallest eigenvalue of that matrix (also known as the Fiedler vector) to divide the network into two communities (depicted by the positive or negative sign of the eigenvalue). Furthermore, those values provide a measure of

affinity towards that group. The closer to zero, the less the affinity of a node towards a group. It was this detail that prompted us to use this method (see Section 3). An example of this algorithm can be seen in Figure 3.

Definition 2.1 *(Graph Laplacian) Let G=(V,E) be a graph on n vertices with edges between nodes i, j, denoted as {i,j}. The graph Laplacian L(G) is the $n \times n$ symmetric integer matrix defined by*

$$f_{ij} = \begin{cases} deg(i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } \{i,j\} \text{ is an edge of } G \\ 0 & \text{otherwise.} \end{cases}$$

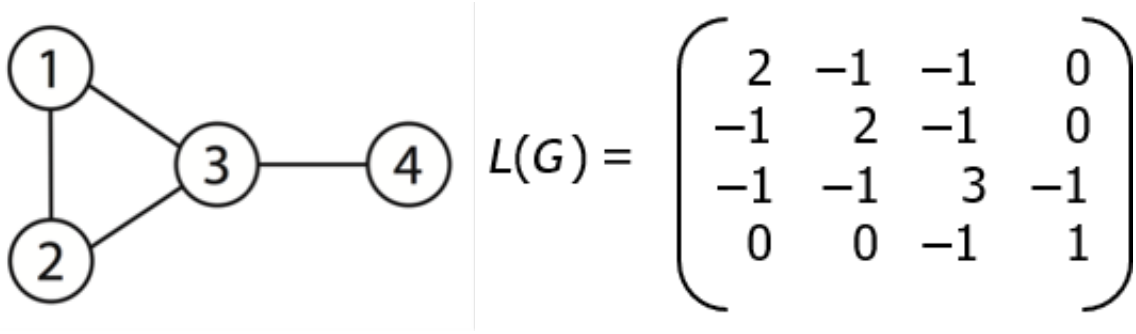


Figure 2: An example of the Graph Laplacian representation of a graph

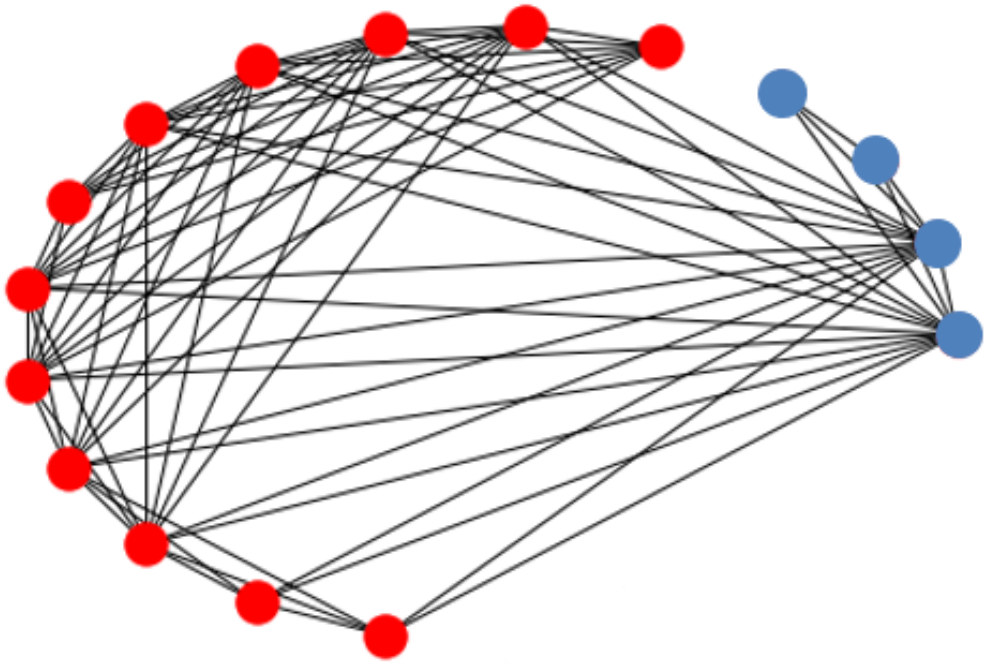


Figure 3: An example of the Spectral Partitioning on the Southern Women's Data Set. The respective communities are represented by color.

2.3 K-Cliques Algorithm

This algorithm is capable of finding an arbitrary number of communities and uses the relationship of nodes and edges in a network to determine if there are communities that exist. An intuitive example is a real-life clique in a social setting, which focuses on whether on all persons in a clique know each other. An individual who knows some members of a clique, but not all, can often be judged as not part of that clique. This method has temporal complexity of $O(n^2k^2)$, where k is the number of edges in the network.

2.4 Fluid Communities

This method, also capable of finding an arbitrary number of communities, is similar to the more familiar Label Propagation Algorithm. Intuitively, this method assigns labels to a subset of nodes in a network and propagates those labels to neighboring nodes to determine the communities. Its time complexity is $O(k)$, and so it is a very quick algorithm, though an disadvantage which is that no unique solution is found, similar to KL. [10].

3 Affinity Score

As mentioned in Section 1.2, one's ability to predict is often dependent on the information one has to describe a network. To predict the community structure of a network, we establish an affinity score to measure the predisposition a node has to be in a particular community. One challenge to predicting community structure is that the current state of a network's communities may not provide much information on what they may later be. For example, if we were to have a network describe individuals in an academic course, just because one student is initially identified to be part of one study group, this does not imply that student may later be part of another group of students. The role this affinity score plays is that it gives us a measure of how inclined a node is to one community or another. In this example of a class of students, an ideal affinity score might tell us how attached a student is to a particular study group, and thereby how likely they may be to stay or leave in that

group. Thus, applying this score to all nodes in a network, it can result in a way to discern in which communities nodes would most likely be a part of.

One would also imagine that over time one might have a better idea of a node's affinity to one group or the other. Thus, a central part of how we use this affinity score to predict communities is accumulating these scores from a time-series analysis. Intuitively, if over time a node shows increasing affinity towards a group, one can reasonably predict that the node will be part of that community at a future point in time.

We explore a variety of ways to extract this kind of affinity score from the networks we work with. We describe them in the following subsections.

3.1 Artificially Generated Propensity

Recall in Section 2.1 that the KL algorithm always begins at a random position in a network. A natural way to take advantage of this randomness is to use a node's propensity for one community or another - determined by running the algorithm several times and then taking the fraction of times when a node was in one community versus the other - as an affinity score. This method of generating an affinity score is computationally expensive, and could only reasonably done with small networks like the Southern Women Data Set (see Section 6.1). However, it was quite simple, as it only required running the algorithm several times and keeping track of how often a node was decided to be in one community or the other.

3.2 Spectral Partitioning

As mentioned in Section 2.2, spectral partitioning provides a natural affinity score with the respective values of the given Fiedler vector, specifically, their proximity to zero. The sign of the value in the Fiedler vector corresponding to a specific node specifies which community that node is in, and that values' distance to zero defines the affinity to the community. For example, if the Fiedler vectors is $\vec{x} = [1, 0.5, -.11, -1]$, then that implies the four nodes the vector represents, in the same order, have strong affinity towards one group, medium affinity to the same group, weak affinity towards the other group, and strong affinity to the other group.

3.3 Centrality-Based Affinity Score

Here, we propose a generalized method to determine an affinity score in the case of a two community partition of a network that uses centrality measures. A centrality measure is a measure to identify the most important nodes in a network. The idea presented here is to use this measure on each community within a network to establish each node's affinity to the network. This method first involves detecting communities in a network at a given point in time, and then calculates the centrality of nodes within those distinct communities. As there are many different kinds of measures, we look at variations in the results between using eigenvector, degree, and betweenness centrality in our method. It is worth noting that the effectiveness of these different kinds of centrality measures likely depends on the inherent structure

of a network. For example, betweenness centrality, which would identify nodes at critical junctures in a network, is likely to be most applicable to settings where the most important figures are mediators or middlemen. Degree centrality looks at nodes that have the most edges, and could help discern the most important figures, for instance, in academic groups. Finally, eigenvector centrality rewards those who have the most connections with other important nodes in a network, and thus could be most helpful to distinguish prominent figures in corporate or political settings.

The resulting affinity score is a tuple (u, v) of a node's community and corresponding centrality measure. This method allows for the use of any community detection algorithm, so long as it accommodates the network itself. In our tests we used Kernighan-Lin, Fluid communities, and k-cliques, though this usage varied where applicable, e.g. computationally possible. For example, the Fluid communities detection method doesn't allow for disconnected graphs.

We also added the measure of the change in centrality a node may have. Intuitively, we tracked the change in affinity a node had from one time period to the next to capture if a node gained affinity towards a community or not. As centrality measures a node's importance to a community, tracking the change in centrality allowed for more information information to enhance our predication capability. It follows that our affinity score was ultimately in the form of a tuple (u, v, w). An algorithm summarizing this process is found below.

Algorithm 1: How to determine centrality based affinity score

Result: Dateframe containing affinity scores

for *t in* DESIRED TIME FRAME **do**

communities = Detected communities in the graph G at time t;

for *c in* COMMUNITIES **do**

center^{i} = arg max *node*$\in$ c ;

for *node in* C **do**

record centrality[center^{c}] - centrality[node]

end

end

save all data for time t;

end

4 Machine Learning Algorithms

In this thesis, we focus on the use of supervised learning. Supervised learning some sort of procedure or model on a portion of the data called the training set, and then uses the rest of the data - the testing data - to verify how effective the fitted model functions. The process of fitting a model to the testing data is described as "learning," hence the term "machine learning." The various machine learning algorithms to compute the predicted communities are Random Forrest, Extra Trees, and XGBoost algorithms which come respectively from Scikit-Learn [11] and XGBoost documentation [12]. All of these methods are decision-tree based methods, but optimize the way they fit onto training data differently. Random Forrest attempts to

calculate an optimal split in a decision tree, whereas Extra Trees assigns random values for these splits, and XGBoost uses a gradient boosting method to determine splits in the decision tree. This variety was chosen to ensure we were not dependent on a single machine learning method, but without unnecessarily complicating the use of our affinity score. Test-train splits were done with a standard 70-30 test-train split.

5 Affinity-Based Prediction

To predict the formation of communities, we first obtain the affinity score as described in Section 3 for each discrete time period of a network, and then use the machine learning algorithms described in Section 4 to carry out the prediction, based off of the time-series affinity scores up to a certain point in time. The time periods were chosen according to the timestamps given in the data, which for our data sets were always by day, except for the Southern Women's Data Set, which was approximately by month. Thus, for each new time period, there is that much more data to predict on. The role of the community detection methods seen in Section 2 were mainly used to determine affinity scores, however, they were also used to identify the final outcome of communities in the networks we explored, as ground truth was not available.

5.1 Benchmarks for Comparison

To understand how well our predictions performed, we considered two different benchmarks. The first, and most simple benchmark was simply using the appropriate community detection method at one point in time, and observing how well that result aligned with the final outcome. In other words, this benchmark analyzed how closely a current state matched the final state. Ultimately, this is the same as seeing how a present outcome matches the final outcome. In the example of a school study group network, this would be akin to just looking how well communities at one point in time match the communities at the final time period. The second benchmark is quite similar to our prediction, but lacks the entire time-series. This benchmark uses the affinity score, but only for a single moment in time. This benchmark helps us also see how well time-series information contributes to our ability to predict the final outcome. Considering the school study group network, this would be equivalent to obtaining the affinity scores at a single point in time and predicting off of that score alone, rather than a cumulative time series.

6 Data

The data sets used to test our methods were selected to provide a wide range of social networks in terms of complexity and size. This was done to verify the robustness of our predictions. It is worth noting that our methods could be used to predict communities in any kind of network, rather than just social networks.

6.1 Southern Women Data Set

This data was from a study in the 1930s [1]. This network has been analyzed by many researchers over the years. It follows a group of 18 women and tracks their interactions and participation in community events (see Figure 1). This was also originally a bipartite, undirected graph, with 89 edges representing the participation of each woman to an event. Thus, we converted this network into a network that shows when women participated at the same event.

6.2 Facebook Data

From the website Konect [13], a small sample of Facebook users has been collected, tracking the friending activities of various individuals. This data represented a very disconnected, unweighted, undirected, network with 63,731 nodes representing users and 817,035 edges representing friendships. To overcome this sparsity and size, we focused at a large connected component of the network at the final time period available, and worked backwards to obtain a proper time-series data set.

6.3 MIT Reality Mining

Also provided by Konect [14], the MIT Reality Mining data set is a study tracking the interactions of 100 MIT students over nine months. Here, two students are connected in the resulting network if they contacted each other in person. This network contains 96 nodes representing persons with 1,086,404 edges representing

contacts, and is an undirected, multiple unweighted graph. The connections in this network are tracked on a daily basis as well, and resulted in our most complex network. Similar to our other networks, this network is disconnected at first, but as time went on became a fully connected network.

6.4 Haggle

This Konect data set [15], similar to the MIT Reality Mining experiment, tracked the phone activity of approximately 270 individuals, seeing when different individuals contacted each other. Like the MIT Reality Mining Data, it is an undirected, multiple unweighted network with 274 nodes and 28,244 edges representing contacts. A unique feature of this data set is that it was fully connected at the beginning, and thus allows us to see how our methods work on a network that does not prominently feature network growth unlike in the other networks.

7 Results

In this section we show the effectiveness of the methods described in the previous section on the various data sets selected (see Section 2). In our figures, the blue line represents the rudimentary benchmark of seeing how well communities at a certain point in time match with the final outcome. The orange line represents the accuracy of our predictions based on cumulative, time-series affinity scores. The green line is the accuracy of predictions based on momentary affinity scores at a single

point in time.

When discussing results for the centrality-based methods as discussed in Section 3.4, as most results were more or less similar, only distinct results showing distinctively different patterns are displayed. A full collection of the figures can be found in the appendix.

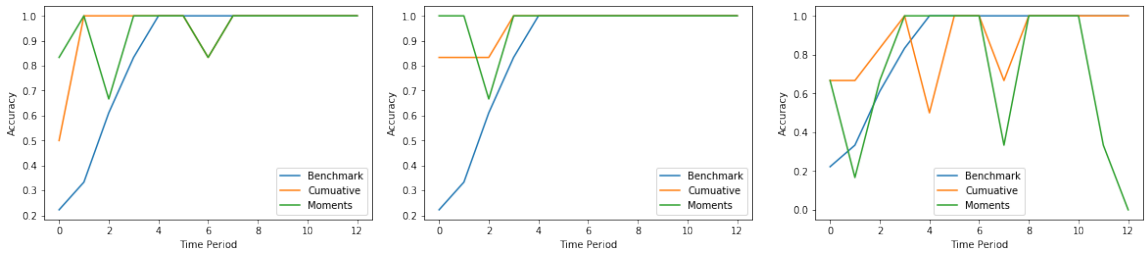
7.1 Southern Women Data Set

In this section we describe how well our methods predicted community formation on the Southern Women Data Set. The following sections do the same for their respective data sets.

7.1.1 Kernighan-Lin

With the generated propensity score (the fraction from repeating the algorithm several times) from the Kernighan-Lin Algorithm, we seem to predict relatively well, regardless of the algorithm, though XGBoost does seem more variable than others (see Figure 3). It is possible that the random nature of the Kernighan-Lin Algorithm does not align well with the gradient boosting methods of XGBoost. An interesting thing to note are the small dips, when a prediction didn't do very well against the final outcome. This is likely due to a possibly large shift in the network structure, or some other update that would have thrown our predictions astray. In summary, it seems that our cumulative method performs well, but not significantly better compared to

other methods. One potential reason for this would be how the Southern Women's Data Set could really consist of multiple groups, rather than two.



Random Forrest Extra Trees XGBoost

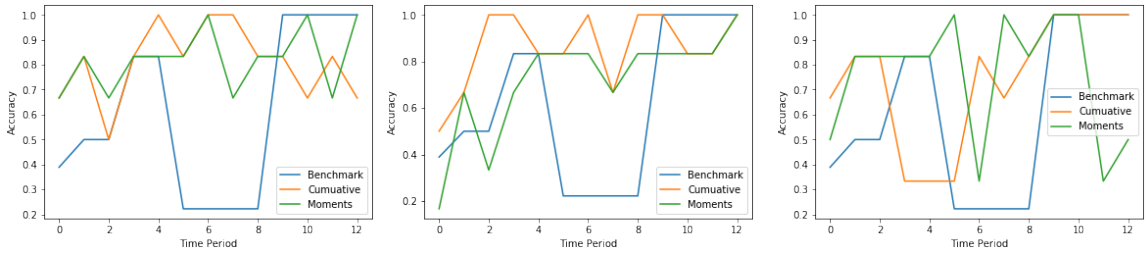
Figure 4: Southern Women Data Set: Kernighan-Lin Bisection

This affinity score was obtained by repeatedly running the Kernighan-Lin algorithm on the network, and taking a node's propensity towards one community or the other

Graphs are labeled by the machine learning algorithm used to produce the prediction

7.1.2 Spectral Partitioning

When using the affinity score derived from spectral partitioning, the results are more noisy than when using Kernighan-Lin. When run with Random Forrest or XGBoost, it is not clear that our prediction does particularly well compared to either benchmark. However, with Extra Trees our prediction seems to consistently perform better than these benchmarks. One advantage of Extra Trees is given how it randomizes the splits in a decision tree, it can often still do well where other methods fail to be effective.



Random Forrest Extra Trees XGBoost

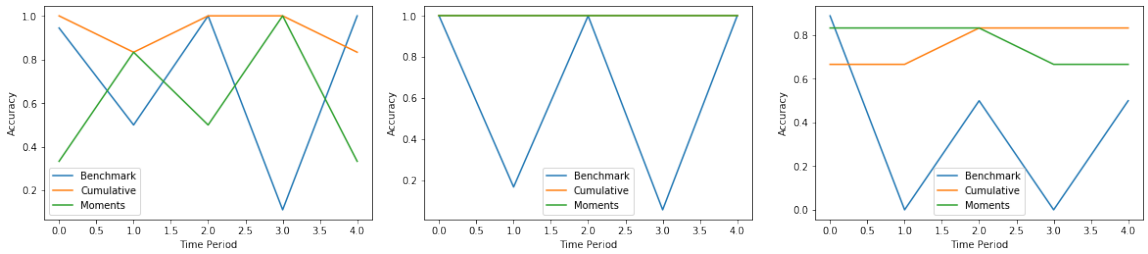
Figure 5: Southern Women Data Set: Spectral Partitioning

This affinity score was obtained by using the values in the eigenvector corresponding the the graph Laplacian representation of the network

7.1.3 Centrality-Based Affinity Score

Testing the centrality-based affinity score on this first network yielded unexpected results. The results varied most depending on the original community detection algorithm, though the choice of centrality measure also had an effect. As we see in Figure 5, our method (represented by the orange line) performs quite well. The constant line at the top of the second graph in Figure 5 demonstrates perfect predictive power, though this was only because there was only one community predicted at all. It is difficult to discern why this is the case, and thus we opted to test further with different kinds of data to better flush out this idea of a centrality-based affinity score. One plausible idea could be in the setting of tracking how 18 women interact, communities form around the most ”important” characters, which is what our centrality-based affinity score is intended to measure. This may also explain why this method seems to do better than what we saw with the Kernighan-Lin propensity score and spectral partitioning. This initial test gives us a positive outlook as to the

usefulness of this predictive method based off of nodes' centrality measures in their respective communities.



Kernighan-Lin	K-Cliques	Fluid Communities
Betweenness Centrality	Eigenvector Centrality	Degree Centraltity
XGBoost	Extra Trees	Extra Trees

Figure 6: Southern Women Data Set: Centrality-Based Affinity Score

This affinity score was obtained by using a tuple of a centrality measure and which community a node currently belonged to.

7.2 Facebook Sample

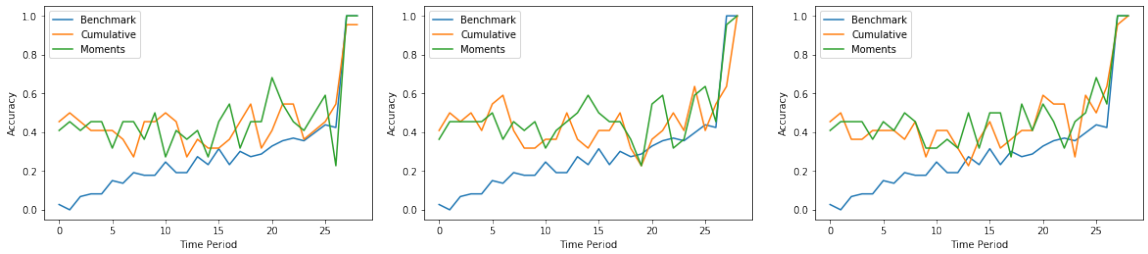
7.2.1 Kernighan-Lin

We remind the reader for reasons explained in Section 3.2, we do not test the Kernighan-Lin algorithm method on these next data sets. Running the algorithm repeatedly was too computationally intense to try and determine a node's likelihood to be in one community or another.

7.2.2 Spectral Partitioning

Predicting communities using the affinity score from spectral partitioning seems to do well in general. However, adding in a time-series aspect does not seem to add

much more to our predictive power than does simply learning on individual moments. Note that when compared to the accuracy we observed in the previous section, overall, we are less accurate until the final time periods. However, it is worth noting that the accuracy spikes up at the very end, which might imply that this network's community structure may have suddenly changed at the end of the time period studied. It is difficult to know the exact circumstances of this network, but one possible explanation for why the green and orange lines which represent both the cumulative time-series and momentary measure are so intertwined is that Facebook friend groups are not prone to much change. It follows that spectral partitioning may not be an ideal community detection method for at least our Facebook sample.



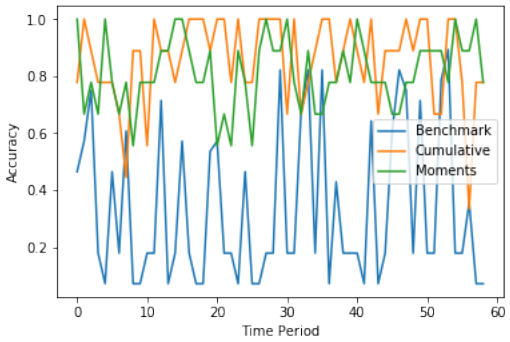
Random Forrest Extra Trees XGBoost

Figure 7: Facebook Sample Data Set: Spectral Partitioning

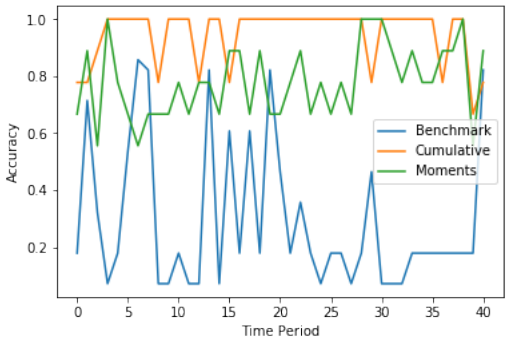
7.2.3 Centrality-Based Affinity Score

Observing the use of the centrality-based affinity score, we see relatively mixed results with using a cumulative time-series, or even the score at all. In some cases, like in Figure 8.b, we see that using a cumulative time-series analysis works well, but in Figure 8.c we note that even the most rudimentary benchmark beats both the moment and cumulative predictions. One will also note in Figure 8.d that we

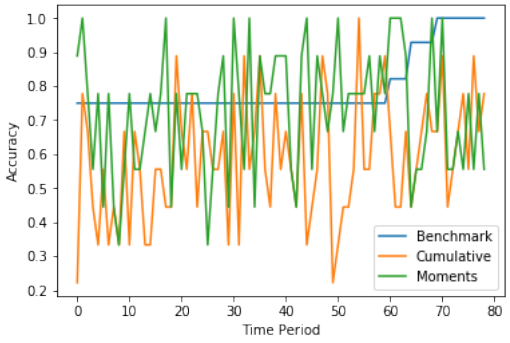
run into the same problem with the Southern women data set, where we perpetually only predicted one community. With the Facebook data set, it seems that we can predict decently using this affinity score, but the time-series factor does not give us substantially better results. In a mass of several users, there may not really be a "most important" figure and so this centrality-based affinity score may not have as much information. Nonetheless, we do see a fair amount of variance in our results, depending on the community detection method and centrality measure we opt to use.



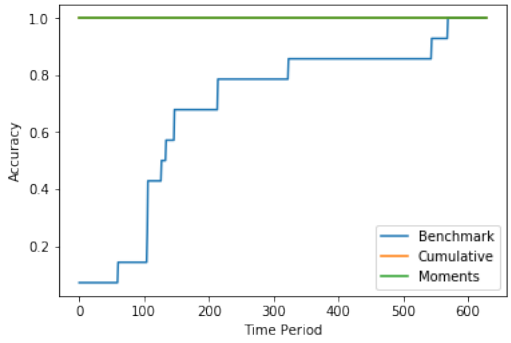
(a) Kernighan-Lin
Degree Centrality
Random Forrest



(b) Kernighan-Lin
Betweeness Centrality
Extra Trees



(c) K-Clique
Eigenvector Centrality
Random Forrest



(d) K-Clique
Betweeness Centrality
XGBoost

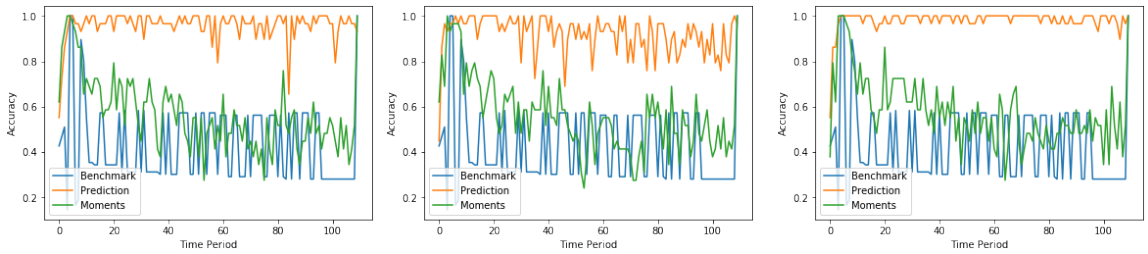
Figure 8: Facebook Sample Data Set: Centrality-Based Affinity Score

Graphs are labeled by the community detection method, centrality measure, and machine learning method used to make predictions

7.3 MIT Reality Mining

7.3.1 Spectral Partitioning

The use of an affinity score based on spectral partitioning works quite well with the MIT data set, as seen in figure 8. Adding in a cumulative time-series gives even more favorable results as time goes on. It is interesting to note how the predictions based on the moments, non-cumulative data, was actually quite poor. In most cases we've seen this prediction to do fairly well, and this is one of the few cases where it hardly beat the rudimentary benchmark. As this network focuses on how individuals contacted each other, it makes sense why the time-series analysis performed so well, as day-to-day contacts probably have little information compared to accumulated contact.



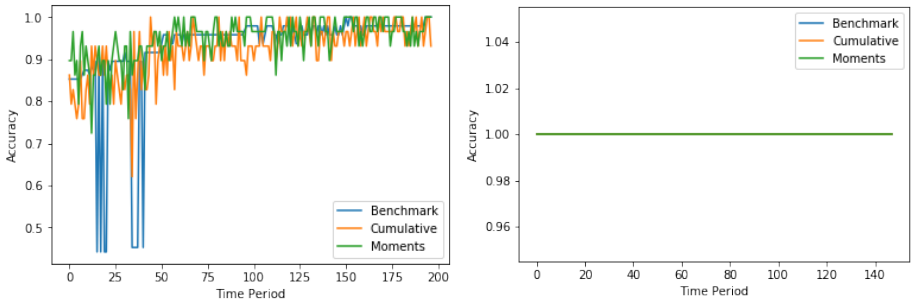
Random Forrest Extra Trees XGBoost

Figure 9: MIT Reality Mining Data Set: Spectral Partitioning

7.3.2 Centrality-Based Affinity Score

With our centrality-based affinity score, the results are far less distinct, and no noticable advantage exists with predicting based on an aggregate series of affinity scores versus just those of one period in time. The K-cliques community detection

algorithm continues to behave very strangely, predicting the entire network to be a single community. This network that tracked when individuals contacted each other likely did not benefit from central figures, as interactions may have been done on a more individual basis and were independently motivated. This may be why the centrality-based affinity score did not yield much effectiveness.



Kernighan-Lin
Eigenvector Centrality
Random Forrest

K-Cliques
Degree Centrality
Extra Trees

Figure 10: MIT Reality Mining Data Set: Centrality-Based Affinity Score

7.4 Haggle Dataset

With the relatively mixed results from both the Facebook and Reality Mining data sets while using our centrality-based method, it seemed more evidence was needed on the effectiveness of both the affinity score and the use of accumulating time-series data. Therefore, we introduce the Haggle data set, previously mentioned in Subsection 6.4. We also recall that both the Facebook and MIT networks were constantly growing as well, and the Haggle data set does not. Only edges were added to the graph. We were interested in seeing how predicting communities might change in such a setting.

7.4.1 Centrality-Based Affinity Score

As we can see in Figure 11, except for the case where we used the k-cliques community detection, our predictions do quite well when detecting communities with either eigenvector or betweenness centrality. With degree centrality in general, the results are not particularly impressive, perhaps because degree centrality is only a local measure of the network. It may also be attributed simply to the intrinsic characteristics of the Haggle network. There is some variance amongst the machine learning algorithms we use, but the overall pattern is the same. Our predictor, as shown in orange, tends to perform more effectively than our benchmark scores seen in blue and green. Apparently, there may have been important figures in this network that motivated interaction, which may be why we say successful predictions on our part. Contact between persons in this network was not independent, and thus using this concept of centrality proved useful.

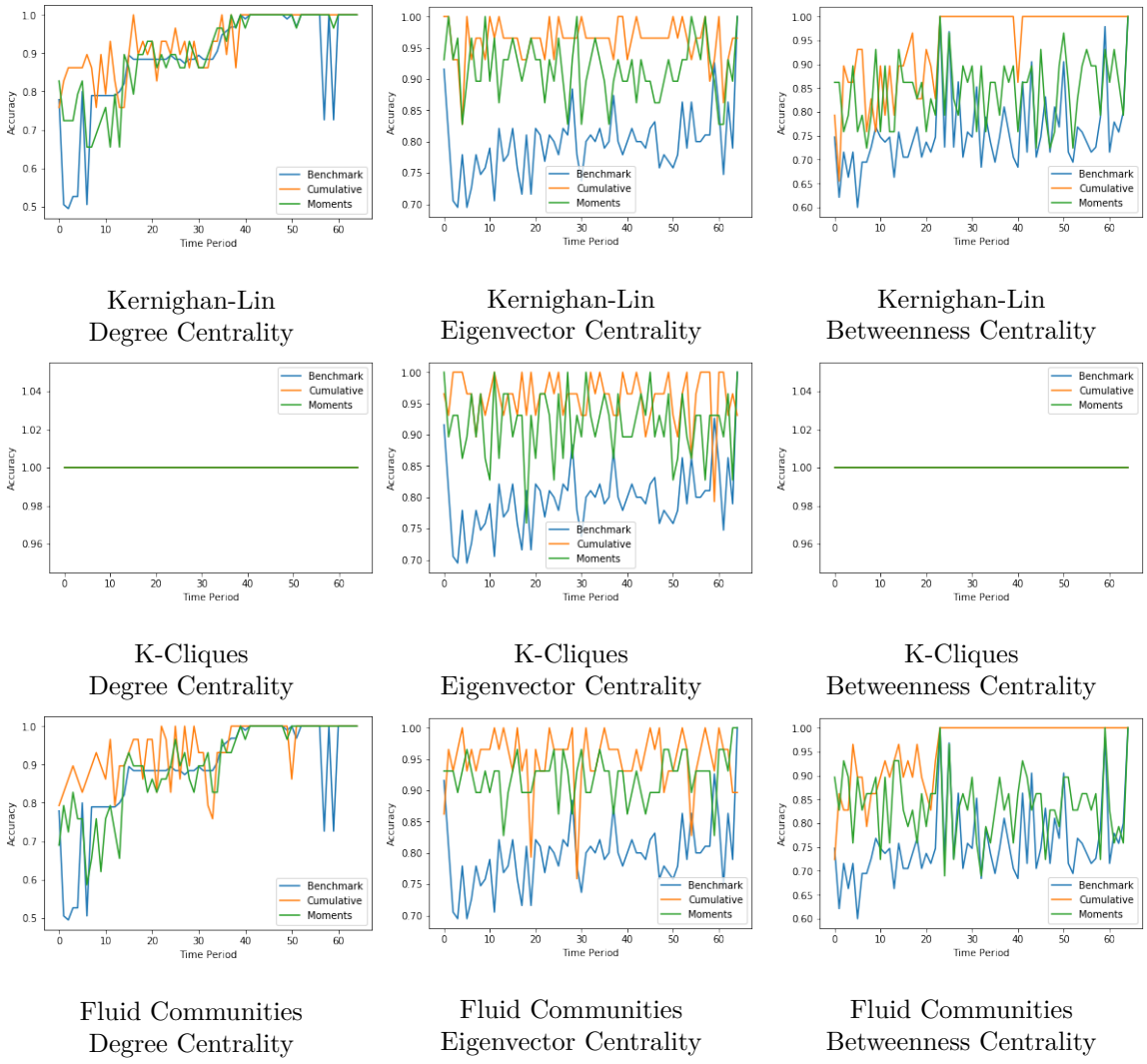


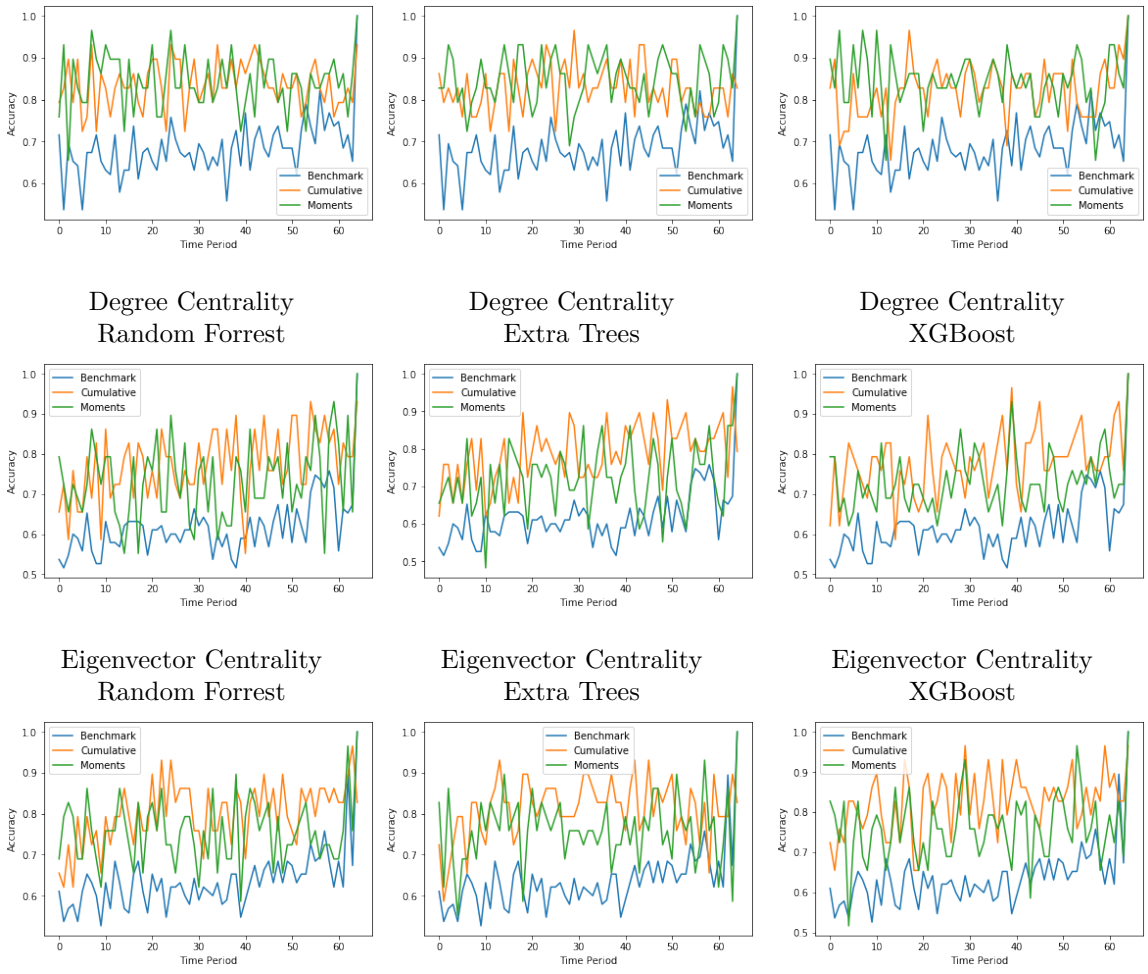
Figure 11: Haggle: Centrality-Based Affinity Score

Graphs are labeled by the community detection algorithm and centrality measure. These predictions were are performed with XGBoost

7.5 Predicting Multiple Communities

With the Haggle dataset, we also tried predict more than just two communities. We only tested multiple communities on the Haggle data set as only the fluid communities community detection method seemed to provide noteworthy results on

any of our data sets, and that method only works on connected graphs. Thus, with the methods we explored, it was not feasible to try and detect more communities with the Facebook and MIT networks. As seen in Figure 11 in the appendix, we have positive results in this regard as well in a similar vein to how we predicted with just two communities. This provides encouraging results as to the generalizability of this method.



Degree Centrality Random Forrest | Degree Centrality Extra Trees | Degree Centrality XGBoost

Eigenvector Centrality Random Forrest | Eigenvector Centrality Extra Trees | Eigenvector Centrality XGBoost

Betweeness Centrality Random Forrest | Betweeness Centrality Extra Trees | Betweeness Centrality XGBoost

Figure 12: Fluid Communities (3 Communities) on the Haggle Data Set

8 Discussion

Here we discuss other things to keep in mind in light of the results we have found, and other potential considerations to move forward with this topic.

8.1 Considerations

A number of limitations exist with our tests in this thesis. One of which is how we did not have robust networks to test on where we knew the ground truth of the structure of communities. Our approach allows us to predict based on a community detection method's results on the final time period allowing us to circumvent this somewhat, but is still a substitute for the actual state of a network. In future studies and tests we would want to work with data sets where we possess the ground truth, despite the rarity of such data.

Another consideration with our centrality-based affinity score is how the combination of centrality measure and community detection method need to be catered towards different kinds of networks. As we see in the results section, some combinations work better than others depending on the network we used. We currently do not know how to best select this kind of combination, nor why that kind of combination works best for that network. It should also be noted that there are many kinds of community detection methods that we did not try to explore.

A final consideration is the limited computing power that I have access to. Network are often very complex and to fully explore them would require computational

power that is often not accessible.

8.2 Further Research

We feel optimistic that much future research can be done regarding this project, especially in our own original centrality-based affinity score. By exploring new combinations of community detection methods and different centrality scores, there is much to be seen. There are also networks of different topics, such as informational or biological, that are worth exploring. Ultimately, we feel that this thesis can help encourage further study into the prediction of communities within complex networks.

9 Conclusion

We find that there are multiple ways to predict communities, though the success of our prediction can often vary much depending on the approach. In this paper we establish a few different ways to tackle this problem, and demonstrate success that we hope encourages future study into this problem.

A Code

A.1 Function to Determine Centrality-Based Affinity Score

```
def community(df):



    #Convert the given data into a networkx graph format

    G = nx.to_numpy_matrix(nx.from_pandas_edgelist(df, 'start', 'end')
        ↪ .to_undirected())




    #choose the community detection method desired

    coms = list(nx.community.asyn_fluidc(nx.Graph(G),3))

# coms = list(nx.community.kernighan_lin_bisection(nx.Graph(G)))

# coms = list(nx.community.k_clique_communities(nx.Graph(G), 3))



    #Record information for each new community

    lilg = [list(coms[i]) for i in range(len(coms))]

    newgraphs = [G[j].T[j] for j in lilg]

    ngs = [nx.Graph(n) for n in newgraphs]



    #Choose the centrality measure desired and record information for
        ↪ the respective new communities
```

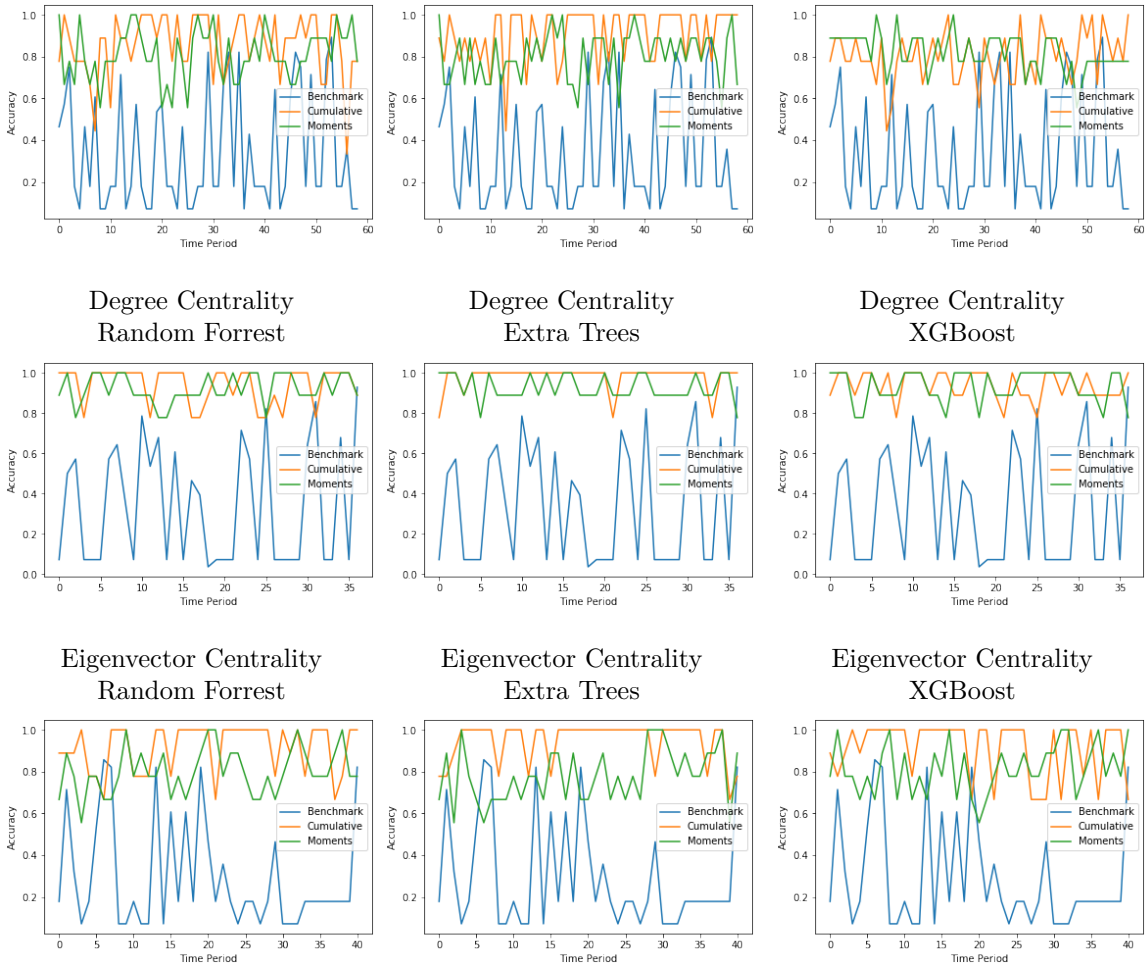
```
# centralities = [nx.eigenvector_centrality(ng).values() for ng in ngs
    ↪ ]
# centralities = [nx.degree_centrality(ng).values() for ng in ngs]
    centralities = [nx.betweenness_centrality(ng).values() for ng in
        ↪ ngs]


    #Compile data
    grps = [[v[np.argmax(centralities[c])] for g in range(len(v))] for
        ↪  c,v in enumerate(lilg)]
    zips = [list(zip(lilg[x], centralities[x], grps[x])) for x in
        ↪ range(len(lilg)) ]
    dl = []
    for z in zips:
        dl.extend(z)
    czip = np.array(sorted(dl))
    values = list(czip[:,1])
    grouping = list(czip[:,2])


    return values, grouping
```

B Full List of Figures Relating to the Centrality-Based Affinity Score

B.1 Results for Facebook Data

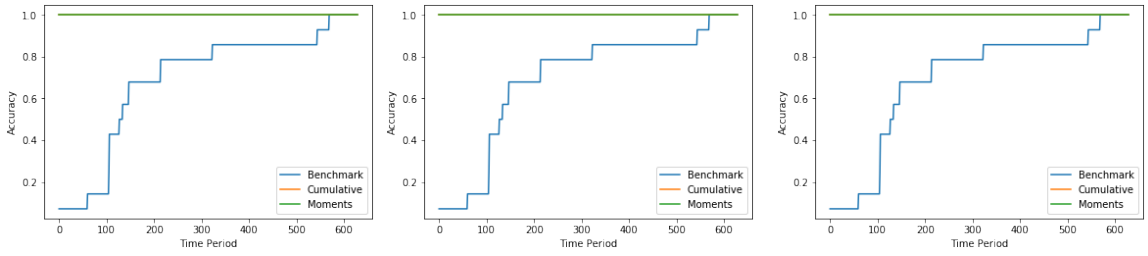


Degree Centrality Random Forrest | Degree Centrality Extra Trees | Degree Centrality XGBoost

Eigenvector Centrality Random Forrest | Eigenvector Centrality Extra Trees | Eigenvector Centrality XGBoost

Betweeness Centrality Random Forrest | Betweeness Centrality Extra Trees | Betweeness Centrality XGBoost

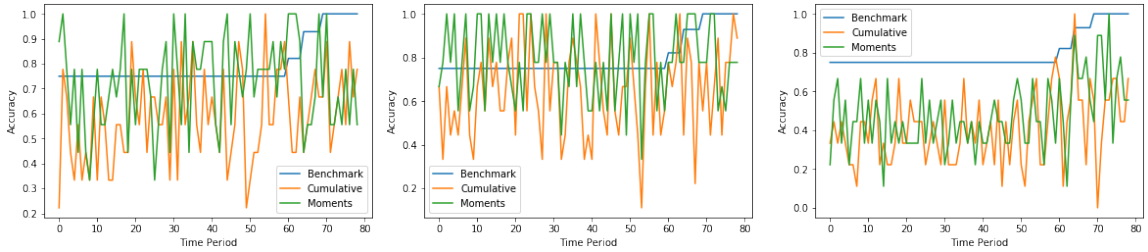
Figure 13: Kernighan-Lin Bisection



Degree Centrality
Random Forrest

Degree Centrality
Extra Trees

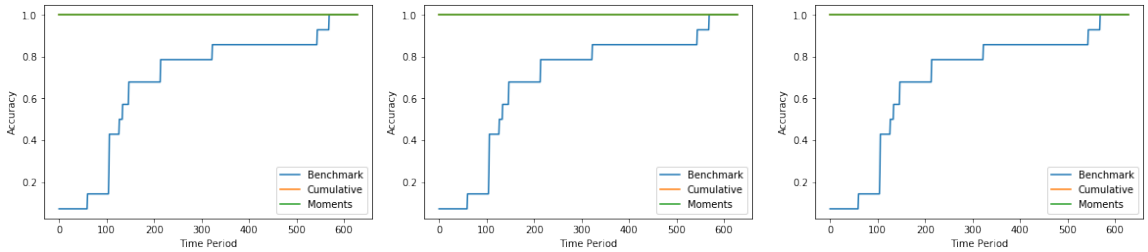
Degree Centrality
XGBoost



Eigenvector Centrality
Random Forrest

Eigenvector Centrality
Extra Trees

Eigenvector Centrality
XGBoost



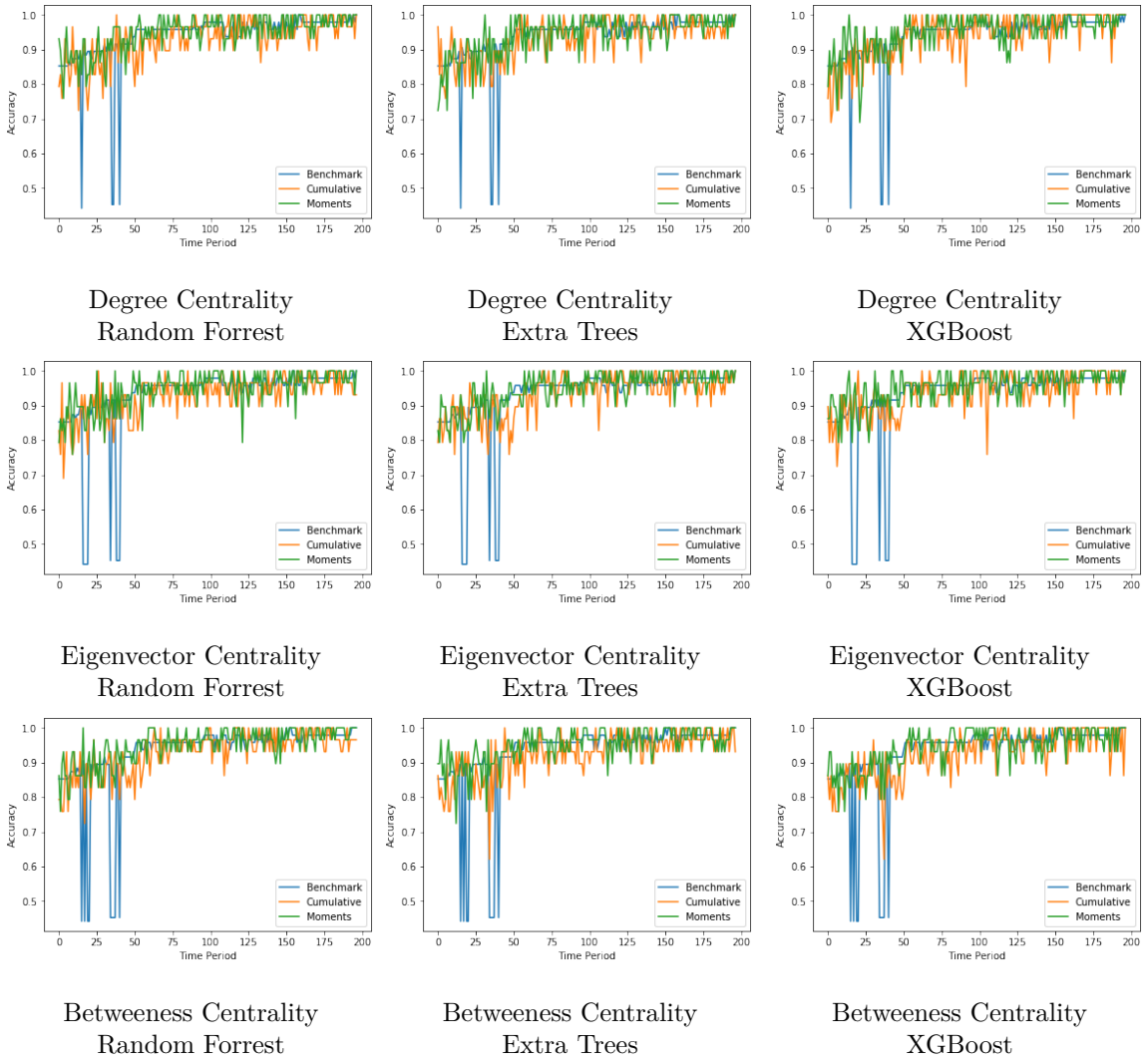
Betweeness Centrality
Random Forrest

Betweeness Centrality
Extra Trees

Betweeness Centrality
XGBoost

Figure 14: K-Clique Neighbors

B.2 Results for MIT Data

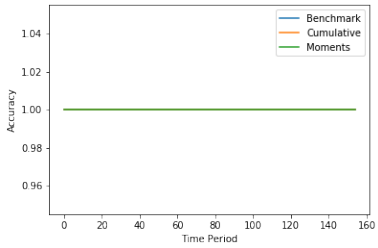


Degree Centrality Random Forrest | Degree Centrality Extra Trees | Degree Centrality XGBoost

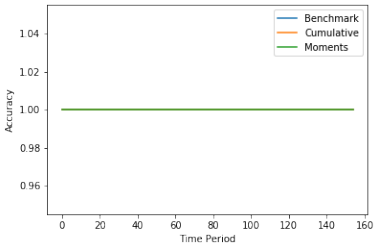
Eigenvector Centrality Random Forrest | Eigenvector Centrality Extra Trees | Eigenvector Centrality XGBoost

Betweeness Centrality Random Forrest | Betweeness Centrality Extra Trees | Betweeness Centrality XGBoost

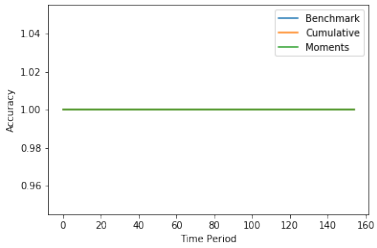
Figure 15: Kernighan-Lin Bisection



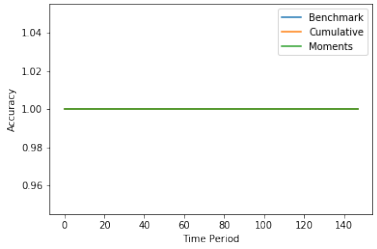
Degree Centrality
Random Forrest



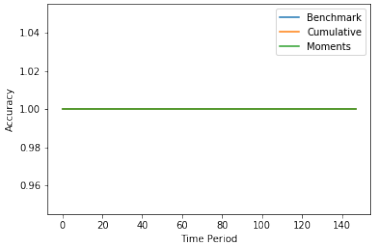
Degree Centrality
Extra Trees



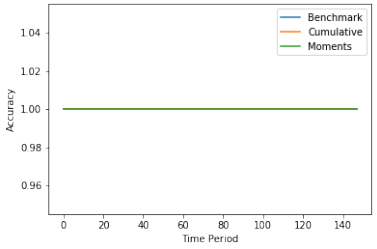
Degree Centrality
XGBoost



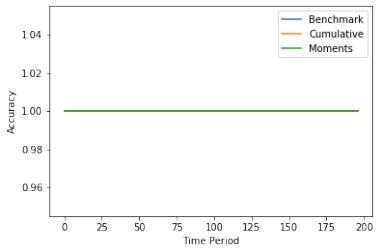
Eigenvector Centrality
Random Forrest



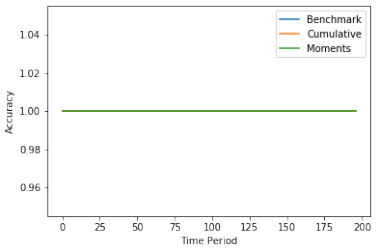
Eigenvector Centrality
Extra Trees



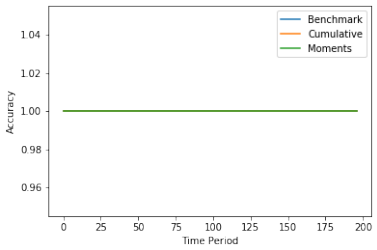
Eigenvector Centrality
XGBoost



Betweeness Centrality
Random Forrest



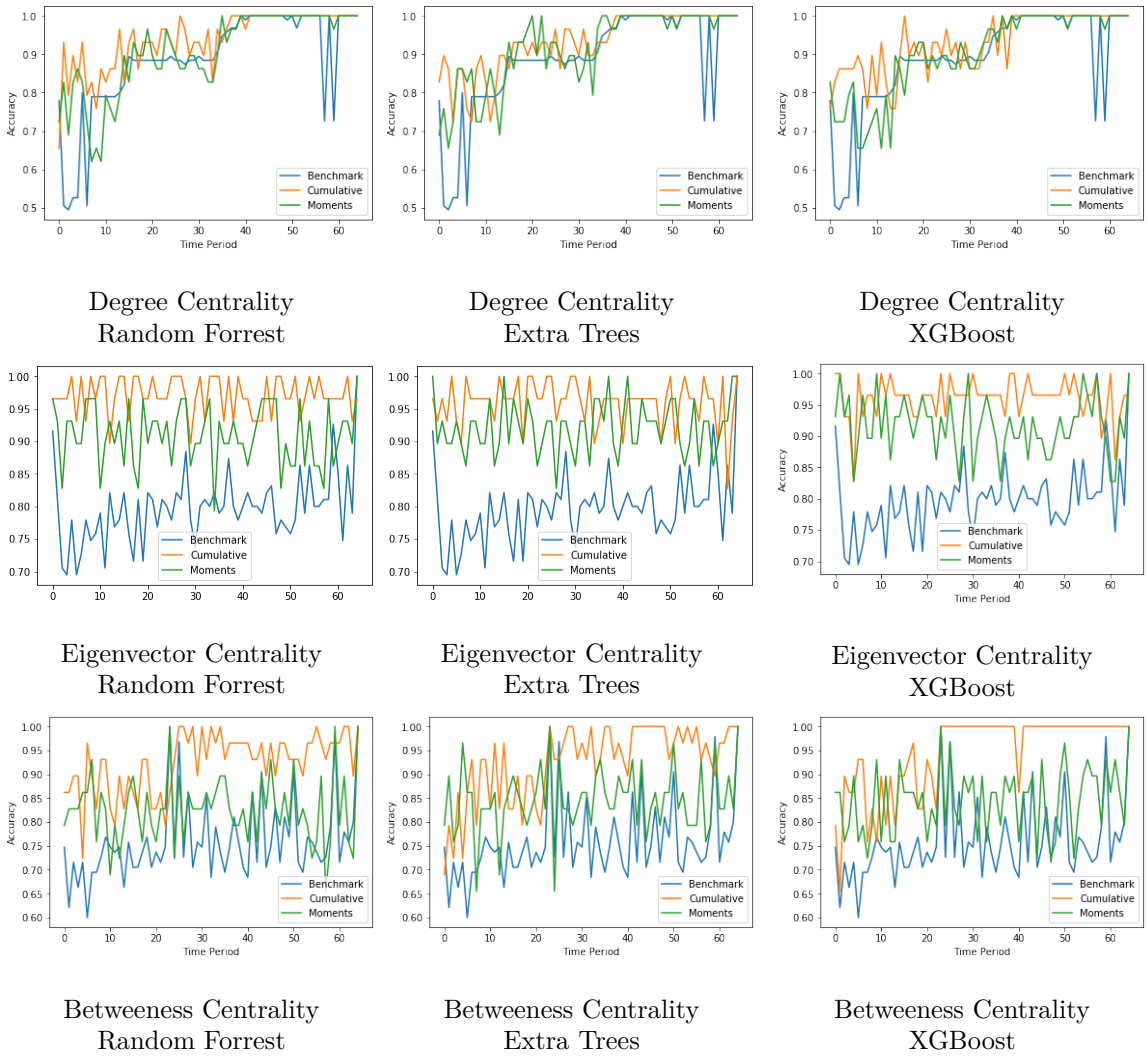
Betweeness Centrality
Extra Trees



Betweeness Centrality
XGBoost

Figure 16: K-Cliques

B.3 Results for Haggle Data

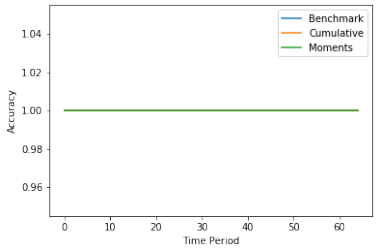


Degree Centrality Random Forrest | Degree Centrality Extra Trees | Degree Centrality XGBoost

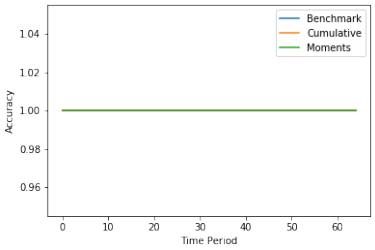
Eigenvector Centrality Random Forrest | Eigenvector Centrality Extra Trees | Eigenvector Centrality XGBoost

Betweeness Centrality Random Forrest | Betweeness Centrality Extra Trees | Betweeness Centrality XGBoost

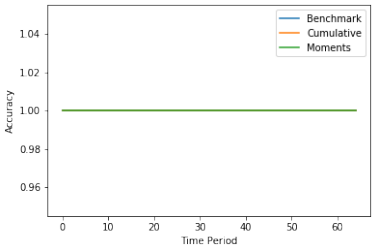
Figure 17: Kernighan-Lin Bisection



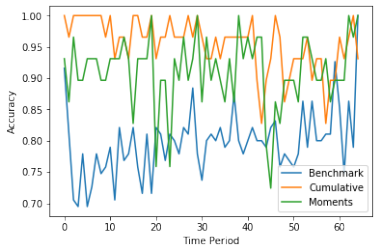
Degree Centrality
Random Forrest



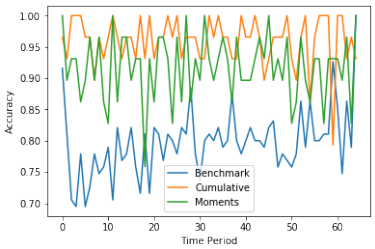
Degree Centrality
Extra Trees



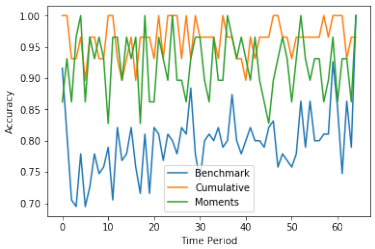
Degree Centrality
XGBoost



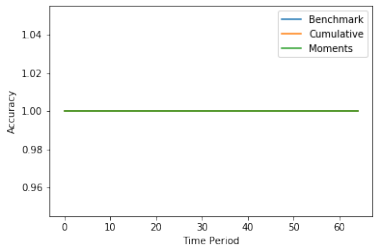
Eigenvector Centrality
Random Forrest



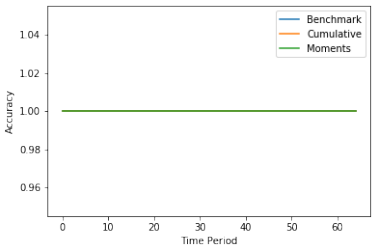
Eigenvector Centrality
Extra Trees



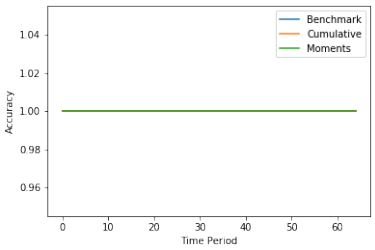
Eigenvector Centrality
XGBoost



Betweeness Centrality
Random Forrest

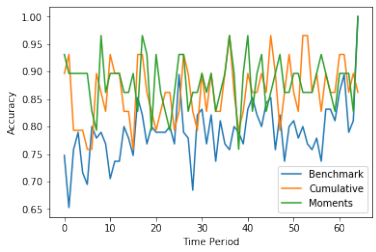


Betweeness Centrality
Extra Trees

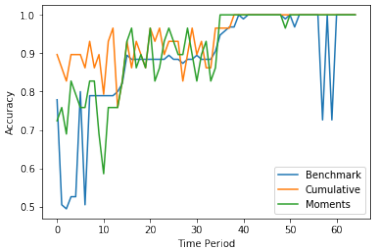


Betweeness Centrality
XGBoost

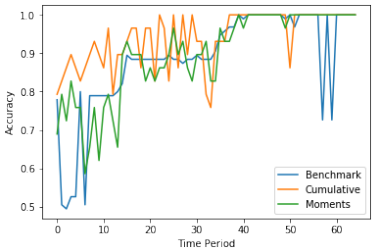
Figure 18: K-Cliques



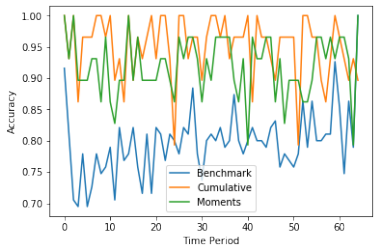
Degree Centrality
Random Forrest



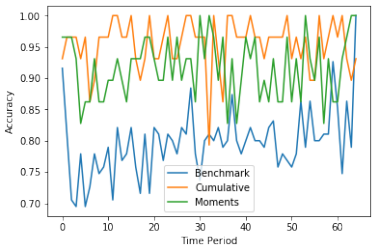
Degree Centrality
Extra Trees



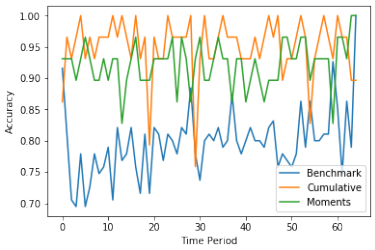
Degree Centrality
XGBoost



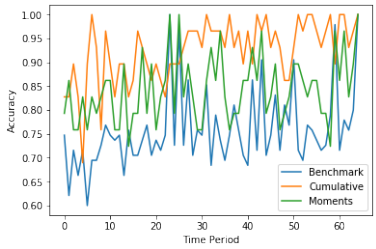
Eigenvector Centrality
Random Forrest



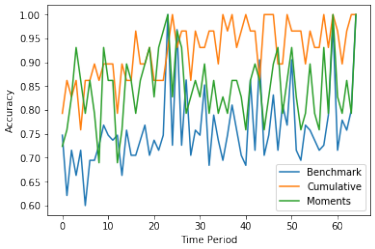
Eigenvector Centrality
Extra Trees



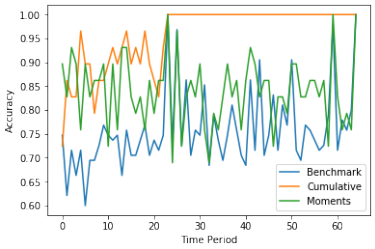
Eigenvector Centrality
XGBoost



Betweeness Centrality
Random Forrest



Betweeness Centrality
Extra Trees



Betweeness Centrality
XGBoost

Figure 19: Fluid Communities

References

[1] Allison Davis, Burleigh Bradford Gardner, and Mary R Gardner. *Deep South: A social anthropological study of caste and class*. Univ of South Carolina Press, 2009.

[2] Ricard V Solé, Sergi Valverde, and Carlos Rodriguez-Caso. Convergent evolutionary paths in biological and technological networks. *Evolution: Education and Outreach*, 4(3):415, 2011.

[3] National Research Council. *Network Science*. The National Academies Press, Washington, DC, 2005.

[4] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[5] Muhammad Aqib Javed, Muhammad Shahzad Younis, Siddique Latif, Junaid Qadir, and Adeel Baig. Community detection in networks: A multidisciplinary review. *Journal of Network and Computer Applications*, 108:87–111, 2018.

[6] Punam Bedi and Chhavi Sharma. Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 6(3):115–135, 2016.

[7] 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.

[8] 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.

[9] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., USA, 2010.

[10] Ferran Parés, Dario Garcia-Gasulla, Armand Vilalta, Jonathan Moreno, Eduard Ayguadé, Jesús Labarta, Ulises Cortés, and Toyotaro Suzumura. Fluid communities: A community detection algorithm. *CoRR*, abs/1703.09307, 2017.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[12] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

[13] Facebook friendships network dataset – KONECT, April 2017.

[14] Reality mining network dataset – KONECT, April 2017.

[15] Haggle network dataset – KONECT, April 2017.