THE COMPLEX NETWORK ANALYSIS OF THE EDUCATION NETWORK OF EMPLOYEES

(ÇALIŞANLARIN EĞITIM AĞININ KARMAŞIK AĞ ANALIZI)

by

CEYDA KOCAMAN, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

SMART SYSTEMS ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

AUGUST 2023

Approval of the thesis:

THE COMPLEX NETWORK ANALYSIS OF THE EDUCATION NETWORK OF EMPLOYEES

submitted by **CEYDA KOCAMAN** in partial fulfillment of the requirements for the degree of **Master of Science in Smart Systems Engineering Department, Galatasaray University** by,

Examining Committee Members:

Assist. Prof. Dr. GÜNCE KEZİBAN ORMAN
Supervisor, **Computer Engineering Department, GSU** _____

Assist. Prof. Dr. REİS BURAK ARSLAN
Computer Engineering Department, GSU _____

Assist. Prof. Dr. BAHRİ ATAY ÖZGÖVDE
Computer Engineering Department, BOUN _____

Date: _____

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Assist. Prof. Dr. Günce Keziban Orman, for providing me with all kinds of knowledge, encouragement, and guidance during both the coursework and thesis phases of my master's education. Throughout my entire academic journey and the thesis period, my family has never withheld their support from me, and I am truly grateful to them. Additionally, I would like to extend my thanks to Softtech for their cooperation during the thesis topic selection process and their support in data acquisition.

July 2023

Ceyda KOCAMAN

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The advances in technology and data science affect many fields positively. One of these fields is education. Learning analytics have the potential to develop new ways of achieving excellence in teaching and learning. The companies try to use learning analytics techniques for their employees' education and aim to improve employee performance. The education data sets of Softtech employees are used in this study. Softtech is a software company in Turkey, and those data sets include different types of technical, non-technical, online, and offline education. All data sets are combined, and an employee network is created by connecting employees via education. In this study, complex network analysis, dynamic and static network analysis, link prediction, and machine learning techniques are applied with the aim of creating an education recommendation system.

The study initially began with community detection in dynamic networks. During the application phase, it was concluded that due to the changes caused by the impact of the COVID-19 pandemic in the data, it was not suitable for community detection in dynamic networks. Therefore, it was decided to proceed with the link prediction method in the study. But the network is extensively examined as a dynamic network.

In method section, all the methods used in the study are explained in detail. Firstly, information about how dynamic and static networks are generated is provided. Subsequently, information regarding the analysis of all micro, macro, meso, and time series required for the analysis of a complex network is shared. Descriptions of all the parameters used in the analyses are provided. After the information about network generation and analysis, a literature review on link prediction methods is shared. Afterwards, information regarding machine learning algorithms to be used for link prediction and performance evaluation metrics for these algorithms are shared. Finally, the method stage is completed by creating a template for machine learning-based link prediction for the education network of employees.

When it comes to the application part, dynamic networks are firstly created. These

dynamic networks are analyzed using the methods described in the method section, and the results are shared. There is data available for a period of 5 years, from January 2017 to December 2021. These data sets are separated on a monthly basis to create dynamic networks. At the end, a total of 59 monthly network files are obtained. Networks are generated from each network file. The connections between employees in relation to their education and the evolution of these connections are visualized on a monthly basis. Visuals from the year 2017 are shared to provide an idea. Sequentially, macro, micro, and meso analyses are conducted for each network. Based on these analysis results, the node counts, link counts, network structures, centrality measures, and community structures of each network are visualized. Information such as the node with the highest degree, the node with the highest betweenness centrality, and the nodes belonging to the same community can be accessed from these visualizations. Some of this information is shared to provide an idea. In the time series analysis section, each parameter is examined over time, and it is checked whether predictions can be made for the future. The impact of the COVID-19 pandemic is evident here. During the quarantine period, all employees work from home, resulting in an increase in the number of employees receiving education (node count) and the number of education sessions (link count) during the COVID-19 pandemic. Due to the lack of a regular distribution, successful results are not observed in terms of prediction. Additionally, the sudden decrease in the number of nodes (employees receiving education) in August 2018 indicates a period of mandatory collective leave.

After analyzing and examining the data, the link prediction process is initiated. After explaining the raw data, the newly created education dictionary, and the newly generated static network structure, link prediction methods are applied in sequence. A total of 15 different link prediction methods are applied. Data until 2021 is used as training data, and data for 2021 is used as test data. These operations are performed using both training and test data. The output file contains 15 attributes corresponding to 15 methods. The output file also includes a label attribute indicating whether there is a connection between nodes in the network. Some of the connection prediction results are shared for illustrative purposes. Ultimately, two separate output files are generated, one for the training data and one for the test data, with 21 columns and 222778 rows each.

The output files generated in the previous step are used as input files in the machine learning step. Prediction is being attempted using machine learning algorithms such as XGBoost, gradient boosting, random forest, logistic regression, support vector machines, and multilayer perceptron. Accuracy, precision, recall, and F1 score values are being examined to compare the accuracy of the models. The results of these values are shared in tables. Different preferences can be made for the model, but generally, support vector machine seems to be more preferred due to its consistency. All of its values are greater than 98%.

The results indicate that the machine learning-based connection prediction method can be used in our study. It is observed that it is applicable to our data, and the results are usable. The subject of how to improve it is addressed. In the next stage, education information can also be recorded on the connections. The strength of the connection can be considered based on the number of shared education received. It is possible to advance and improve the study with different methods.

Keywords : COMPLEX NETWORK ANALYSIS, DYNAMIC AND STATIC NETWORKS, LINK PREDICTION, MACHINE LEARNING, EDUCATION RECOMENDATION, LEARNING ANALYTICS

ÖZET

Teknoloji ve veri bilimindeki gelişmeler birçok alanı olumlu yönde etkilemektedir. Bu alanlardan biri de eğitimdir. Öğrenme analitiği, öğretme ve öğrenmede mükemmelliğe ulaşmanın yeni yollarını geliştirme potansiyeline sahiptir. Şirketler, çalışanlarının eğitimi için öğrenme analitiği tekniklerini kullanmaya çalışmakta ve çalışan performansını artırmayı hedeflemektedir. Bu çalışmada Softtech çalışanlarının eğitim veri setleri kullanılmıştır. Softtech, Türkiye'de bir yazılım şirketidir ve bu veri setleri farklı teknik, teknik olmayan, çevrimiçi ve çevrimdışı eğitim türlerini içerir. Tüm veri setleri birleştirilir ve çalışanlar eğitimler aracılığıyla birbirine bağlanarak bir çalışan ağı oluşturulur. Bu çalışmada bir eğitim öneri sistemi oluşturmak amacıyla karmaşık ağ analizi, dinamik ve statik ağ analizi, bağlantı tahmini ve makine öğrenmesi teknikleri uygulanır.

Çalışma, başlangıçta dinamik ağlarda topluluk tespiti ile başladı. Uygulama aşamasında gelindiğinde, verilerde COVID-19 pandemisinin etkisinin neden olduğu değişiklikler nedeniyle dinamik ağlarda topluluk tespiti için uygun olmadığı sonucuna varıldı. Bu nedenle çalışmada bağlantı tahmin yöntemi ile ilerlemeye karar verildi. Ancak ağ, kapsamlı bir şekilde dinamik bir ağ olarak incelenir.

Yöntem bölümünde çalışmada kullanılan tüm yöntemler detaylı bir şekilde anlatılmıştır. Öncelikle dinamik ve statik ağların nasıl oluşturulduğu hakkında bilgi verilmektedir. Ardından karmaşık bir ağın analizi için gerekli olan tüm mikro, makro, mezo ve zaman serilerinin analizine ilişkin bilgiler paylaşılmaktadır. Analizlerde kullanılan tüm parametrelerin açıklamaları verilmiştir. Ağların oluşturulması ve analizi hakkında bilgi verildikten sonra bağlantı tahmin yöntemleri ile ilgili literatür taraması paylaşılmıştır. Daha sonra bağlantı tahmini için kullanılacak makine öğrenmesi algoritmaları ve bu algoritmalara ait performans değerlendirme metrikleri hakkında bilgiler paylaşılmıştır. Son olarak, çalışanların eğitim ağı için makine öğrenimi tabanlı bağlantı tahmini konusunda bir şablon oluşturularak yöntem aşaması tamamlanır.

Uygulama kısmına gelindiğinde ise öncelikle dinamik ağlar oluşturulmaktadır. Bu dinamik ağlar, yöntem bölümünde açıklanan yöntemlerle analiz edilir ve sonuçlar paylaşılır. Ocak 2017'den Aralık 2021'e kadar 5 yıllık bir döneme ait veriler mevcuttur. Bu veri kümeleri aylık olarak ayrıştırılarak dinamik ağlar oluşturulur. Sonunda toplam 59 aylık ağ dosyası elde edilir. Ağlar, her ağ dosyasından oluşturulur. Çalışanların eğitim ile birbirlerine nasıl bağlandıkları ve bu bağlantıların değişimi aylık olarak görselleştirilir. 2017 yılına ait görseller fikir vermesi amacıyla paylaşılır. Sırasıyla, her ağ için makro, mikro ve mezo analizler yapılır. Bu analiz sonuçlarına dayanarak, her bir ağın düğüm sayıları, bağlantı sayıları, ağ yapıları, merkeziyet ölçüleri ve topluluk yapıları görselleştirilir. Derecesi en yüksek düğüm, arasındalık merkeziliği en yüksek düğüm, aynı topluluğa ait düğümler gibi bilgilere bu görsellerden ulaşılabilir. Bu bilgilerin bir kısmı fikir vermesi amacıyla paylaşılır. Zaman serisi analizi bölümünde her bir parametre zaman içinde incelenir ve geleceğe yönelik öngörülerde bulunup bulunamayacağı kontrol edilir. COVID-19 pandemisinin etkisi burada açıkça görülür. Karantina döneminde tüm çalışanların evden çalışması, COVID-19 pandemisi sırasında eğitim alan çalışan sayısında (düğüm sayısı) ve eğitim sayısında (bağlantı sayısı) artışa neden olur. Düzenli bir dağılımın olmadığı için tahmin konusunda başarılı sonuçlar görülmez. Ayrıca, 2018 yılının Ağustos ayındaki düğüm (eğitim alan çalışanlar) sayısındaki ani düşüş, zorunlu toplu izin dönemine işaret etmektedir.

Verileri analiz edip inceledikten sonra bağlantı tahmin süreci başlatılır. Ham veriler, yeni oluşturulan eğitim sözlüğü ve yeni oluşturulan statik ağ yapısı açıklandıktan sonra, bağlantı tahmin yöntemleri sırayla uygulanır. Toplam 15 farklı bağlantı tahmin yöntemi uygulanmaktadır. 2021 yılına kadar olan veriler eğitim verisi, 2021 yılı verisi ise test verisi olarak kullanılmaktadır. Bu işlemler hem eğitim hem de test verileri kullanılarak gerçekleştirilir. Çıktı dosyası, 15 yönteme karşılık gelen 15 öznitelik içerir. Çıktı dosyası ayrıca ağdaki düğümler arasında bir bağlantı olup olmadığını gösteren bir etiket özniteliği içerir. Bağlantı tahmini sonuçlarından bazıları fikir vermesi açısından paylaşılmıştır. En nihayetinde, her biri 21 sütun ve 222778 satır içeren, biri eğitim verileri ve diğeri test verileri için olmak üzere iki ayrı çıktı dosyası oluşturulur.

Bir önceki adımda oluşturulan çıktı dosyaları, makine öğrenimi adımında girdi dosyaları olarak kullanılır. XGBoost, gradyan artırma, rastgele orman, lojistik regresyon, destek vektör makineleri ve çok katmanlı algılayıcılar gibi makine öğrenimi algoritmaları kullanılarak tahmin yapılmaya çalışılır. Doğruluk, kesinlik, hatırlama ve F1 puanı değerleri incelenerek modellerin doğruluğu karşılaştırılır. Bu değerlerin sonuçları

tablolarda paylaşılır. Model için farklı tercihler yapılabilir ancak genel olarak tutarlılığı dolayısıyla destek vektör makinesi daha tercih edilebilir görünüyor. Tüm değerleri %98'den büyüktür.

Makine öğrenmesi tabanlı bağlantı tahmin yönteminin çalışmamızda kullanılabileceği sonucu ortaya çıkmıştır. Verimize uygulanabilir olduğu ve sonuçların kullanılabilir olduğu görülmektedir. Nasıl geliştirilebilir konusu ele alınmıştır. Sonraki aşamada bağlantılara eğitim bilgileri de kaydedilebilir. Bağlantının gücü, alınan ortak eğitim sayısına göre değerlendirilebilir. Çalışmayı daha farklı yöntemlerle ilerletmek ve geliştirmek mümkündür.

Anahtar Kelimeler : KARMAŞIK AĞ ANALİZİ, DİNAMİK VE STATİK AĞLAR, BAĞLANTI TAHMİNİ, MAKİNE ÖĞRENMESİ, EĞİTİM TAVSİYESİ, ÖĞRENME ANALİTİKLERİ

1 INTRODUCTION

Data science technologies are used in the field of human resources, just like in many other areas. Human resources departments should be able to respond to issues such as the risk of employees leaving work, who will be employed, who will be educated, what will be the return of education, and so on with data analytics. Basically, data-oriented suggestions should be provided for problems related to human resource functions. Big data analysis will become increasingly important, especially in organizations where the number of employees is very large. Companies that see human resources as the company's most important strategic investment have to take advantage of data science technologies to oversee human resources processes and make decisions based on data. Data science technologies are rapidly adopted by companies that want to achieve the highest efficiency level in their human resources departments. It is critical that the human resources departments be able to act on the basis of data and concrete output, sign applications that will make a difference, and influence their finances and strategies. It is particularly more important in countries with a large young population. If a merit-based system is adopted in business life, the country will develop along with its institutions. The return on investment in data science in the field of human resources will be more than sufficient.

With advancements in technology and data science, new developments in the field of education are occurring every day. The concept of learning analytics was formed in the education sector through studies conducted in the field of data analytics. Data science studies in the education field increase day by day, and simultaneously, studies in learning analytics that aim to improve more effective ways of learning and teaching increase too. Learning analytics is used to personalize the learning process by processing big data, enabling the design of learning processes based on learners' learning needs, behaviors, and emerging patterns. It is used in creating predictive models related to the future. In the world, numerous applications are being made in the field of learning analytics, and promising results are being obtained. The subject of learning analytics is a field that offers many opportunities for research and development studies to be conducted on it. It is possible to study on and implement it for every age group. Human resources departments also make efforts to acquire knowledge and implement

learning analytics for the training of employees.

Adult learning is a part of learning analytics. As a result of the fourth industrial revolution, adults are constantly trying to update their knowledge and skills about their jobs. Adapting to the changing nature of work, 21st-century skills, new technologies, etc. has critical importance for people. Because of changing global conditions, businesses support their employees' education. Working adults continue their lifelong learning journey in a variety of ways, including through online education, graduate-level education, company education, and so on. Online education facilitates the lifelong learning process for adults, especially during the COVID-19 pandemic era. In that process, the volume of data related to online education increases rapidly. That situation has supported the growth of the learning analytics and educational data mining communities. Many studies about learning analytics have been carried out to date, and it still provides an opportunity for new studies. Untapped potential exists, in particular, for understanding how adult learners navigate their learning experience across course options and over time (Tatel et al., 2022).

In technical part of the learning analytics, it is possible to deal with many methods : data visualization, educational data mining, recommendation systems, forecasting, personalized adaptive learning, social network analysis, etc. The main aim has to be solving a specific problem, and so it will be easier to choose method and get results effectively.

Indeed, the analysis of the collected data from online education resources is a challenging task because, first, it involves describing the formal problems with the effectiveness of the education and, second, it involves finding the appropriate methodology for the analysis. Among several different analysis techniques, such as supervised or unsupervised learning of tabular data sets, natural language processing, or video/sound analysis, network science is emerging with the richness of graph modeling for representing objects' interactions. Network science has benefited from the advanced computational capabilities and increased availability of digital data. Network analysis has greatly contributed to the emergence and advancement of learning analytics, and this is not surprising. Because network analysis facilitates the analysis of teaching and learning with computational, analytical, and representational support, it offers a suite of methods to analyze learning and learners. Many different types of data about learning can be examined by using network approaches, for example, social interactions in forums, friendship ties in the classroom, co-enrollment in courses, and so on. There are

opportunities and challenges to strengthen future work on situating network-analytical research within learning analytics (Chen and Poquet, 2022).

This study began with the desire and idea of conducting a thesis on learning analytics in human resources management. Therefore, we focused on adult learning. In our study, we decided to treat the data as a complex network and analyze it. In previous studies on link prediction in complex network analysis, education data was not utilized. Therefore, in order to contribute something new to the literature, we proceeded with the link prediction technique. A framework for an education recommendation system has been developed. At the end of the study, the thesis was concluded by providing guidance on future work.

In this study, it is focused on treating and analyzing the education data set as a network and modeling it. There are multiple ways to model education data, but better data modeling techniques are needed to understand the hidden interests of employees in their education preferences. For this issue, network modeling is employed. The education data sets of Softtech employees are used in this study. Softtech is a software engineering company in Turkey, and those data sets include different types of technical, non-technical, online, and offline education. Detailed information about the data will be provided in the subsequent sections of the study. Other information about Softtech is available on its website (softtech.com.tr, n.d.). Networks are generated by associating employees who have received the same education with each other. Firstly, the network is considered a dynamic network. Macro analysis, micro analysis, meso analysis, and time-series analysis methods are applied, and the results are obtained. Then, the network is considered a static network, and link prediction techniques and machine learning techniques are applied. An effective and accurate education recommendation system is important in terms of enhancing the employee experience and benefiting the company's economy. With the aim of building an education recommendation system, the problem of education recommendation is posed, thereby targeting the prediction of new potential connections in the employee-education interaction system. Even if employees do not know each other and there is no closeness between them, they can be influenced by each other because they are part of the same system, and these hidden influences can affect their preferences. For that analysis, complex network modeling of employee-education interactions is useful. There is no need the personal or detail information of the employees. At the end of the study, the company may have smarter information about employee learning.

After this section, there are four more chapters, which are organized in accordance with the thesis format. In Chapter 2, all methods that are used in application parts are described. That chapter begins with network generation methods, and then it continues with network properties, network analysis, link prediction, and supervised learning methods. It ends with machine learning-based link prediction for the education network of employees. First, information about dynamic networks and static networks is provided. Metrics used in network analysis are explained. Definitions are made for time series analysis and its subcategory, ARIMA. Under the title of link prediction, descriptions and formulas of fifteen different link prediction methods are given. In the supervised learning methods section, machine learning techniques and performance evaluation methods covered in this context are shared. Finally, a detailed explanation of the machine learning-based link prediction method for the education network of employees is provided, including a flowchart. In Chapter 3, detailed information about the data is shared. Data pre-processing processes are explained. Using the methods described in the second section, networks are analyzed, and the results are shared. In Chapter 4, under the title of link prediction, the following stages are included in order : data pre-processing, network generation, link prediction, and machine learning. In Chapter 5, the study is concluded by commenting on all the obtained results. Future works that will be complementary to this study are also proposed.

If it is needed to list the contributions of this thesis study :

- ✓ It is shown with experiments that network modeling is a suitable analytic model for modeling the education networks of employees.
- ✓ The education links before and during the COVID-19 pandemic between employees of Softtech, which is one of the biggest software companies in Turkey, are examined multidimensionally with network modeling, and analytical results are obtained.
- ✓ The education data is modeled for the first time with a dynamic network. The features of these networks are analyzed with time series analysis.
- ✓ The question of which education employees will receive is attempted to be answered through link prediction. In this context, a wide range of link prediction methods and supervised learning methods are used, and the results are evaluated.

2 METHOD

2.1 Network Generation and Network Properties

A network refers to a system comprising interconnected nodes or entities linked by edges or connections. Networks can encompass various systems, including social relationships, computer systems, transport routes, or biological systems. Broadly speaking, networks can be classified into two main categories : static networks and dynamic networks.

Static networks remain relatively unchanged over time, with connections between nodes either remaining constant or changing slowly. Examples of static networks include social networks of friends or road networks connecting different cities. In these networks, nodes and edges may be fixed or undergo only gradual modifications.

On the other hand, dynamic networks experience changes in connections between nodes over time, influenced by dynamic processes. Examples of dynamic networks include communication networks between people over messaging applications or protein interaction networks within cells. In these networks, nodes and edges can undergo constant changes or evolve, making them more intricate to model and analyze.

In the context of this study, the nodes represent employees, and the links between employees represent their educational connections. Essentially, employees are connected to one another based on their educational backgrounds. The direction of the link is not significant, meaning the graph is undirected. Additionally, the number of joint training sessions between two employees is not considered ; having at least one common educational experience is sufficient to establish a connection. Employee information remains concealed for information security purposes. All forms of training, whether online, offline, classroom-based, or virtual, provided by the company or received by employees from different platforms are taken into account. In addition to employee and training information, date information is also utilized. For dynamic network analysis, the data is divided on a monthly basis using the date information, creating monthly networks. For static network analysis, the date information is not crucial and may only be used to separate training and test data.

A network can be analyzed by macro, meso and micro analysis way. In macro analysis, we examine network-wide properties such as **Node Number, Link Number, Density, Transitivity, Average Degree, Diameter, Radius, Connected Component, number of cliques, Betweenness Centralization, Closeness Centralization, Eigenvector Centralization, Degree Centralization**. In micro analysis, we examine node or link level properties. They are **Degree, Betweenness Centrality, Closeness Centrality, Eigenvector Centrality, Degree Centrality, local transitivity etc.**. In meso analysis, we focus on node groups such as communities, cores, k-cores, nclan etc.

In this thesis, we employ following community detection algorithms.

— Edge Betweenness : Community structure detection based on edge betweenness.

— Fast Greedy : Community structure via greedy optimization of modularity.

— Infomap : Infomap community finding.

— Label Propagation : Finding communities based on propagating labels.

— Leading Eigenvector : Community structure detecting based on the leading eigenvector of the community matrix.

— Walktrap : Community strucure via short random walks.

— Louvain : Finding community structure by multi-level optimization of modularity.

2.2 Time Series Analysis

Time series analysis in dynamic network analysis refers to the study of changes in the structure and properties of networks over time. It involves analyzing the evolution of networks and identifying patterns, trends, and correlations in the network dynamics.

Dynamic networks are networks that change over time, either due to the addition or removal of nodes or edges, or changes in the properties of existing nodes or edges. Time series analysis is used to track these changes and understand how they affect the structure and function of the network.

Time series analysis in dynamic network analysis is useful for understanding a wide range of complex systems, including social networks, biological networks, and transportation networks. By analyzing the dynamics of these networks, we can gain insights into

the underlying processes driving their behavior and develop strategies for optimizing their performance.

ARIMA (AutoRegressive Integrated Moving Average) : ARIMA is a time series modeling technique that is commonly used in dynamic network analysis to forecast future values of a network metric. ARIMA models are used to model the dependence of a variable on its own past values and on the past values of the errors, or deviations from the model's predictions.

ARIMA models can be used to model both the temporal and spatial dynamics of network data. In the context of network analysis, ARIMA models can be used to predict the future values of network metrics, such as the number of edges, the degree distribution, or the clustering coefficient.

The ARIMA model is typically represented as (p,d,q), where p is the order of the autoregressive component, d is the degree of differencing (i.e., the number of times the time series is differenced to remove trends or seasonal patterns), and q is the order of the moving average component. The values of p, d, and q are determined by analyzing the autocorrelation and partial autocorrelation functions of the time series data.

2.3 Link Prediction

Link prediction is used to predict the missing connections in a network. These missing connections may be future connections between nodes too. Its main aim is to find possible connections between nodes after analyzing the structure of the current network. These connections may exist in the network or not. It is possible to see link prediction applications in different areas like recommender systems, social networks, biological networks, et al.

Link prediction features are classified in three parts according to their essential techniques : local, global, and embedding.

For a brief introduction before explaining the methods :

— $G = (V, L)$ defines a network.

— V defines node set of the network.

— L defines link set of the network.

— $N(u)$, or (N_u) defines neighborhood of a node.

— $N(u) = \{v \in V : (u, v) \in L\}$ defines the set of nodes directly connected to u.

The methods for link prediction with local information are explained below :

Common Neighbors (CN) : It is the size of the set of common neighbors between any two nodes. If the number of degrees is higher, it is more possible to have higher Common Neighbors for the nodes. Because of that reason, Common Neighbors has a tendency to be high for any two hub nodes (Newman, 2001). The formula of Common Neighbors is shared in Eq. 2.1.

$$s(u, v) = |N_u \cap N_v| \tag{2.1}$$

Adamic Adar (AA) : It penalizes the scores for hub neighbors. In other words, it counts the total number of neighbors of all common neighbors, but depresses the score by a logarithmic function for demoting the scores of higher degree nodes (Adamic and Adar, 2003). The formula of Adamic Adar is shared in Eq. 2.2.

$$s(u, v) = \sum_{i \in N_u \cap N_v} \frac{1}{\log_2 (|N_i|)} \tag{2.2}$$

Resource Allocation (RA) : It is the same with Adamic Adar, but the difference between Resource Allocation and Adamic Adar that Resource Allocation considers the degrees, not their logarithms. In addition, it also counts the total number of neighbors of all common neighbors (Zhou et al., 2009). The formula of Resource Allocation is shared in Eq. 2.3.

$$s(u, v) = \sum_{i \in N_u \cap N_v} \frac{1}{|N_i|} \tag{2.3}$$

Jaccard Coefficient (JC) : It is the ratio of the number of common neighbors to the number of all neighbors of two nodes and was developed for comparing two sets (Jaccard, 1912). The formula of Jaccard Coefficient is shared in Eq. 2.4.

$$s(u,v) = \frac{|N_u \cap N_v|}{|N_u \cup N_v|} \quad (2.4)$$

Sørrenson/Dice Index (Dice) : It measures the common parts of the neighborhoods, normalizes them with the sizes of the neighborhoods of the two studied nodes, and penalizes being a hub as well. Sørrenson/Dice becomes lower than Jaccard Coefficient when the neighborhoods have many nodes in common but also the common neighbors have many other links to the outside of the common neighborhood (Dice, 1945). The formula of Sørrenson/Dice Index is shared in Eq. 2.5.

$$s(u,v) = \frac{2 \cdot |N_u \cap N_v|}{|N_u| + |N_v|} \quad (2.5)$$

Cannistraci-Alanis-Ravasi Index (CAR) : It is the sum of the numbers of common neighbors of two nodes, each having neighbors in common with the other (Cannistraci et al., 2013b). The formula of Cannistraci-Alanis-Ravasi Index is shared in Eq. 2.6.

$$s(u,v) = \sum_{i \in N_u \cap N_v} 1 + \frac{|N_u \cap N_v \cap N_i|}{2} \quad (2.6)$$

CAR-based Adamic and Adar (CAA) : It combines two strategies : favoring clique-like neighborhoods and penalizing being a hub. In other words, it is a combination of the Cannistraci-Alanis-Ravasi and Adamic Adar strategies (Cannistraci et al.,

2013b). The formula of CAR-based Adamic and Adar is shared in Eq. 2.7.

$$s(u,v) = \sum_{i \in N_u \cap N_v} \frac{|N_u \cap N_v \cap N_i|}{\log_2 (N_i)} \quad (2.7)$$

CAR-based Resource Allocation (CRA) : It is another hybrid metric that combines the Cannistraci-Alanis-Ravasi index and Resource Allocation strategies (Cannistraci et al., 2013b). The formula of CAR-based Resource Allocation is shared in Eq. 2.8.

$$s(u,v) = \sum_{i \in N_u \cap N_v} \frac{|N_u \cap N_v \cap N_i|}{|N_i|} \quad (2.8)$$

Preferential Attachment (PA) : It promotes the nodes that have higher degrees and assumes that the famous nodes should have a higher probability of connecting with each other. Shortly, it is the multiplication of degrees of two nodes (Newman, 2001). The formula of Preferential Attachment is shared in Eq. 2.9.

$$s(u,v) = |N_u| \cdot |N_v| \quad (2.9)$$

CAR-based Preferential Attachment (CPA) : It is the combination of Cannistraci-Alanis-Ravasi and preferential attachment strategies (Cannistraci et al., 2013b). The formula of CAR-based Preferential Attachment is shared in Eq. 2.10.

$$s(u,v) = e_u.e_v + e_u.CAR(u,v) + |e_v.CAR(u,v) + CAR(u,v)^2 \quad (2.10)$$

Here, $e_u = |N_u \backslash (N_u \cap N_v)|$ and $e_v = |N_v \backslash (N_u \cap N_v)|$ is the number of the neighbors

that are not common neighbors of u and v, and $CAR(u, v)$ is the CAR score between nodes u and v.

The methods for link prediction with global information are explained below :

L3 Link Predictor (L3) : It considers network paths of length three. The metric applies a degree normalization strategy because the third-level neighbors numbers are exponentially larger than the second-level ones. The biased high scores coming from the hub nodes, which are naturally building shortcuts and increasing the number of third-level neighbors for entire nodes, are also avoided (Kovács et al., 2019). L3 link predictor (L3), considers network paths of length three (Kovács et al., 2019). The formula of L3 Link Predictor is shared in Eq. 2.11.

$$s(u, v) = \sum_{ij} \frac{a_{\mathrm{ui}}.a_{\mathrm{ij}}.a_{\mathrm{jv}}}{\sqrt{k_i.k_j}} \qquad (2.11)$$

Here, a_{ui} is 1 if there is a link between the nodes u and i. And k_i is the degree of node i. Since the third level neighbors numbers are exponentially larger than the second level ones, the metric applies a degree normalization strategy. It also avoids the biased high scores coming from the hub nodes which are naturally building shortcuts and increases the number of third level neighbors for entire nodes.

Structural perturbation method (SPM) : It is a technique that is similar to the first-order perturbation in quantum mechanics ; it focuses on perturbing the adjacency matrix and observing the change in eigenvalues provided the fixed eigenvectors. The scores are produced for all links based on the perturbation of links removed from the adjacency matrix of the original network, basically (Lü et al., 2015).

The methods for link prediction with embedding are explained below :

Isometric mapping (ISOMAP) : Isometric mapping (ISOMAP), uses one of the traditional graph embedding techniques (Kuchaiev et al., 2009). The studied network, $G = (V, L)$, is first transformed to a distance matrix D of its nodes in which each member d_{uv} of D is the shortest distance between the nodes u and v from V. Then D is transformed to a lower dimensional matrix $L \in \mathbb{R}^l$ with Multidimensional scaling based on non-linear embedding method, MDS. Here l is the new dimension that G is

transformed to. MDS tries to keep original distance d_{uv} between the node pairs and generates new vectors $x_1, x_2, ..., x_n$ for each node whose lengths are l. $x_1, x_2, ..., x_n$ is found as a minimizer of some cost function $min_{x_1,x_2,...,x_n}(d_{\text{uv}} - ||x_u - x_v||)^2$. Once MDS generates new lower dimensional vectors for each node, then ISOMAP calculates basic euclidean distance between the nodes as their dissimilarities.

Laplacian Eigenmaps (LEIG) : Firstly, the laplacian matrix of the original network is generated, and then the spectral decomposition of the corresponding laplacian matrix is computed because it uses a minimization function that can be solved by the generalized eigenvalue problem. Laplacian Eigenmaps find l eigenvalues and eigenvectors with l is the number of new dimensions. The link prediction is again done by considering the Euclidean distance of the node pairs after embedding (Belkin and Niyogi, 2002).

Centered and non-centered Minimum Curvilinear Embedding (MCE & ncMCE) : These are two network embedding techniques that use the distances in the minimum spanning trees of studied networks. Firstly, the minimum spanning tree is generated, and then the distances of every pair of nodes in the minimum-spanning tree are computed in both methods. The name of these distances, which are in the form of a distance matrix, is kernel. If the centering is not chosen in the algorithm, an economy-size singular value decomposition of the distance matrix is performed by non-centered Minimum Curvilinear Embedding. Otherwise, an algebraic operation is performed for kernel centering first, and then the decomposition is done. At the end, the new lower-dimensional space is produced by the transposition of the product of the computed singular values with the right singular vectors with the algebraic corrections (Cannistraci et al., 2013a).

2.4 Supervised-Learning Methods

Supervised learning is a method of machine learning in which an algorithm is trained on labeled data to make predictions or decisions. In this approach, the algorithm learns from a set of input-output pairs, called a training set, to make predictions on new, unseen data.

The input-output pairs in the training set are called examples or instances, where each instance consists of an input and the corresponding desired output. During the training phase, the algorithm learns to map the inputs to their corresponding outputs by adjusting its internal parameters until it produces accurate predictions on the training

set.

Once the algorithm is trained, it can be used to make predictions on new, unseen data by applying the learned mapping function to the new input. The performance of a supervised learning algorithm is evaluated by measuring how well it generalizes to new, unseen data, using metrics such as accuracy, precision, recall, and F1 score.

Some examples of supervised learning algorithms include linear regression, logistic regression, decision trees, support vector machines, and neural networks. Supervised learning is widely used in various applications, such as image recognition, speech recognition, natural language processing, and recommendation systems.

2.4.1 Algorithms

XGBoost : XGBoost (eXtreme Gradient Boosting) is a powerful machine learning algorithm that is widely used for supervised learning tasks such as regression, classification, and ranking. It is based on the gradient boosting framework and is designed to be highly efficient and scalable.

XGBoost uses a decision tree ensemble approach to make predictions. It iteratively builds a series of decision trees, where each new tree is trained to correct the errors made by the previous tree. The output of the model is the weighted sum of the predictions made by each individual tree.

One of the key advantages of XGBoost is its ability to handle large and complex datasets. It is optimized for speed and scalability, and can handle datasets with millions or even billions of rows and thousands of features. It also includes a variety of regularization techniques to prevent overfitting and improve generalization performance.

Gradient Boosting : Gradient Boosting is a machine learning technique used for both regression and classification tasks. It is based on the idea of boosting, which combines multiple weak learners to form a strong learner. The basic idea behind Gradient Boosting is to sequentially add models to the ensemble that predict the residuals (or errors) of the previous models, in order to improve the overall prediction accuracy.

Gradient Boosting works by iteratively fitting a weak learning model to the residuals of the previous model. In each iteration, the algorithm calculates the negative gradient

of the loss function with respect to the predictions of the previous model. This gradient is used to fit the new model to the residuals, so that the new model learns to correct the errors made by the previous model.

The key advantage of Gradient Boosting over other boosting techniques is that it can handle a wide variety of loss functions and can be used with any differentiable loss function. This makes it a highly flexible and powerful technique that can be used in many different applications.

Gradient Boosting is a powerful and widely used technique in machine learning, and is particularly effective for problems where there are complex interactions between the features and the target variable, or where the data is noisy or has missing values.

Random Forest : Random forest is a popular machine learning algorithm that belongs to the family of ensemble methods. It is used for both classification and regression tasks. Random forest is built by combining multiple decision trees, where each decision tree is trained on a randomly selected subset of the training data and a randomly selected subset of features.

Random forest is called a "forest" because it consists of a large number of decision trees, where each decision tree is grown independently and in parallel. When making a prediction, the random forest algorithm aggregates the predictions of all the individual decision trees to arrive at a final prediction. This approach helps to improve the accuracy and robustness of the predictions.

One of the advantages of random forest is that it can handle large datasets with high dimensionality, noisy data, and missing values. It also provides a measure of feature importance, which can be useful in understanding the underlying data patterns and identifying the most relevant features for the task at hand.

Logistic Regression : Logistic regression is a popular machine learning algorithm used for binary classification problems, where the goal is to predict one of two possible outcomes (e.g., yes/no, true/false, 0/1). It is a type of regression analysis that models the relationship between a dependent variable (the binary outcome) and one or more independent variables (features).

In logistic regression, the dependent variable is modeled as a function of the independent variables using a logistic or sigmoid function. The output of this function is

a probability value between 0 and 1, which represents the likelihood of the positive outcome (e.g., yes, 1) given the values of the input features.

Logistic regression is often used in situations where the relationship between the independent variables and the dependent variable is not linear. It is also used when the goal is to understand the effect of each independent variable on the outcome, as logistic regression provides coefficients that indicate the strength and direction of the relationship between the variables.

Support Vector Machine : Support Vector Machine (SVM) is a powerful machine learning algorithm used for classification, regression, and outlier detection. SVM aims to find the best boundary (called hyperplane) that separates different classes in the input data by maximizing the margin between the classes.

In binary classification problems, the SVM algorithm tries to find the hyperplane that separates the two classes with the maximum margin. The margin is the distance between the hyperplane and the nearest data points from each class. The SVM algorithm tries to minimize the margin error while maximizing the margin distance.

SVM can handle both linearly separable and non-linearly separable data by using different types of kernels. A kernel function transforms the input data into a higher dimensional space where it becomes linearly separable. SVM can use various kernel functions such as linear, polynomial, radial basis function (RBF), and sigmoid.

Multilayer Perceptron : Multilayer Perceptron (MLP) is a popular machine learning algorithm that is used for supervised learning tasks such as classification and regression. MLP is a type of feedforward artificial neural network that consists of multiple layers of interconnected nodes or neurons.

In MLP, the input is propagated through the network in a forward direction, passing through multiple hidden layers before reaching the output layer. Each neuron in the hidden layers and output layer applies a non-linear activation function to the weighted sum of the inputs it receives from the previous layer. This allows MLP to model complex non-linear relationships between the input and output variables.

MLP uses a backpropagation algorithm to train the weights of the network, which involves iteratively adjusting the weights to minimize the error between the predicted output and the actual output. The backpropagation algorithm computes the gradients

of the error function with respect to the weights of the network and updates the weights accordingly.

2.4.2 Performance Evaluation Metrics

Accuracy : Accuracy is a metric used in machine learning to evaluate the performance of a classification model. It measures the percentage of correctly classified instances out of the total number of instances in the dataset.

The accuracy of a classification model is calculated as in Eq. 2.12 :

$$\text{Accuracy} = \frac{(\text{Number of correctly classified instances})}{(\text{Total number of instances})} \quad (2.12)$$

For example, if a model classifies 90 out of 100 instances correctly, its accuracy would be 90%.

Accuracy is a useful metric for evaluating the overall performance of a model, but it can be misleading in some cases. For example, in datasets with imbalanced class distributions (i.e., when one class is much more common than the other), a model that always predicts the majority class can achieve high accuracy, even though it is not useful in practice. In such cases, other metrics such as precision, recall, and F1 score may be more appropriate for evaluating the model's performance.

Overall, accuracy is a simple and intuitive metric that provides a quick way to assess the performance of a classification model, but it should be used in conjunction with other metrics to get a more complete picture of the model's performance.

Precision : Precision is a metric used in machine learning to evaluate the performance of a classification model. It measures the proportion of true positives (i.e., correctly classified positive instances) out of all instances classified as positive by the model.

The precision of a classification model is calculated as in Eq. 2.13 :

$$\text{Precision} = \frac{(\text{Number of true positives})}{(\text{Number of true positives} + \text{Number of false positives})} \quad (2.13)$$

For example, if a model correctly classifies 80 instances as positive, but also misclassifies 20 negative instances as positive, its precision would be 80% (i.e., 80 / (80 + 20)).

Precision is a useful metric when the cost of false positives (i.e., instances that are wrongly classified as positive) is high. For example, in a medical diagnosis task, false positives may lead to unnecessary and potentially harmful treatments, so high precision is desirable to minimize the number of false positives.

Overall, precision is a valuable metric for evaluating the performance of a classification model, especially in situations where false positives are costly. However, it should be used in conjunction with other metrics such as recall, accuracy, and F1 score to get a more complete picture of the model's performance.

Recall : Recall, also known as sensitivity or true positive rate, is a metric used in machine learning to evaluate the performance of a classification model. It measures the proportion of true positives (i.e., correctly classified positive instances) out of all actual positive instances in the dataset.

The recall of a classification model is calculated as in Eq. 2.14 :

$$\text{Recall} = \frac{(\text{Number of true positives})}{(\text{Number of true positives} + \text{Number of false negatives})} \quad (2.14)$$

For example, if a model correctly identifies 90 out of 100 positive instances in a dataset, but also misses 10 positive instances, its recall would be 90% (i.e., 90 / (90 + 10)).

Recall is a useful metric when the cost of false negatives (i.e., instances that are wrongly classified as negative) is high. For example, in a cancer diagnosis task, false negatives may lead to delayed or missed treatment, so high recall is desirable to minimize the

number of false negatives.

Overall, recall is a valuable metric for evaluating the performance of a classification model, especially in situations where false negatives are costly. However, it should be used in conjunction with other metrics such as precision, accuracy, and F1 score to get a more complete picture of the model's performance.

F1 Score : The F1 score is a metric used in machine learning to evaluate the performance of a classification model. It is the harmonic mean of precision and recall, providing a balanced measure of the two metrics.

The F1 score of a classification model is calculated as in Eq. 2.15 :

$$\text{F1 score} = 2 \cdot \frac{(precision \cdot recall)}{(precision + recall)} \tag{2.15}$$

The F1 score ranges from 0 to 1, with a higher value indicating better performance. A perfect F1 score of 1 means that the model has both perfect precision and recall.

The F1 score is a useful metric when both precision and recall are important and have equal weight. It is commonly used in binary classification problems where there is an imbalance between the number of positive and negative instances in the dataset.

For example, in a medical diagnosis task, a high F1 score would indicate that the model is able to correctly identify the positive instances (i.e., patients with the disease) while minimizing false positives and false negatives.

Overall, the F1 score is a valuable metric for evaluating the performance of a classification model, especially when both precision and recall are important. However, it should be used in conjunction with other metrics such as accuracy, precision, and recall to get a more complete picture of the model's performance.

2.5 Machine Learning-Based Link Prediction for The Education Network of Employees

Employees have hidden interests in the education they prefer and receive. To understand these interests, there is a need for better data modeling than traditional methods. For this issue, network modeling is employed. An effective and accurate education recommendation system is important in terms of enhancing the employee experience and benefiting the company's economy. In this study, the main aim is to build an education recommendation system. With the aim of building a good education recommendation system, the problem of education recommendation is posed, thereby targeting the prediction of new potential connections in the employee-education interaction system. Even if employees do not know each other and there is no closeness between them, they can be influenced by each other because they are parts of the same system, and these hidden influences can affect their preferences. For that analysis, complex network modeling of employee-education interactions is useful. The data set is handled as a static network. Data pre-processing, link prediction, and machine learning procedures are applied in order.

The link prediction is one of the sub-domains of network science and it is used for finding missing links in networks. It is possible to divide the hundreds of different link prediction approaches into three parts. In this study, machine-learning-based methods were preferred among the three categories. One of the other categories is traditional methods, and the last one is graph embedding techniques that have appeared in recent works. The details of the experiment that involves link prediction via machine learning can be seen in Fig. 2.1.

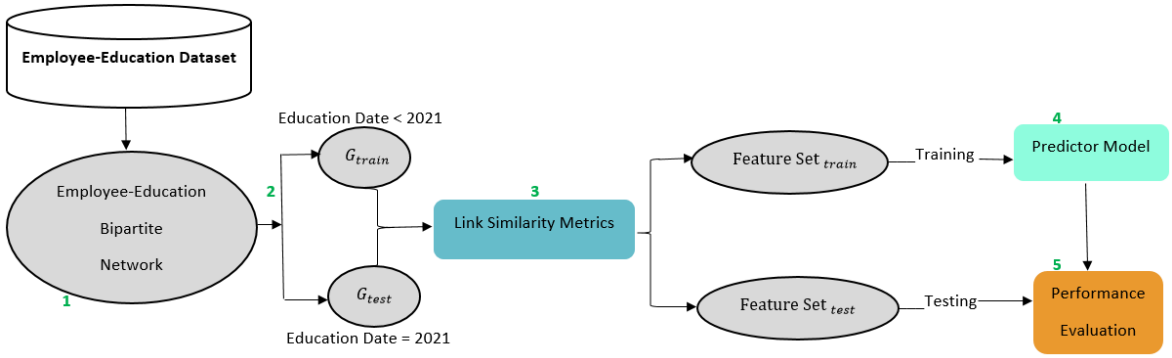


FIGURE 2.1 – Flowchart of the framework that is generated for machine learning-based link prediction for the education network of employees

The main data set is converted to a bipartite network, and then it is split into two different auxiliary projected networks by filtering the bipartite network by education date between employee-education pairs. If the education date is before 2021, it is training data, or in other words, the Gtrain projected network. After splitting the network into a training network and a test network, all link prediction features are calculated. This process is applied to all possible links. This operation includes both appearing and missing links for each network separately. FeatureSettrain and FeatureSettest are crated for Gtrain and Gtest projected networks, respectively. While creating these sets, a label column is created according to whether the link was seen or not. If the link already exists in the network, the value of the label is 1. Otherwise, the value of the label is 0. FeatureSettrain and FeatureSettest are used as a training data set and a test data set in machine learning algorithms, respectively. The results are evaluated by using different machine learning algorithms like XGBoost, gradient boosting, random forest, support vector machine, logistic regression, and multilayer perceptron. Accuracy, precision, recall, and F1 scores are used for evaluating the performance of the machine learning models.

3 NETWORK GENERATION RESULTS

3.1 Study on Dynamic Networks

In this part of the study, education data sets of employees are used. Mainly, there are two different data sets. One of them is an online education data set ; the other is an offline and virtual class education data set. This data set includes all education from January 2017 to November 2021.

There are 669 employees ; their names are masked and changed to user IDs. The columns in "Online Education Data Set" are user ID, education name, sub-education name, and end date of the education. The columns in the "Offline and Virtual Class Education Data Set" are user ID, education name, sub-education name, start date of the education, and end date of the education. There are 24774 rows in the "Online Education Data Set" and 16439 rows in the "Offline and Virtual Class Education Data Set".

After data pre-processing processes, the data set is handled as a complex network. Complex network analysis methods are used. It is treated as a dynamic network rather than a static network. The R programming language has been used in all studies so far.

3.1.1 Data Pre-processing

The two main data sets "Online Education Data Set" and "Offline and Virtual Class Education Data Set," are combined into one data set that is named "Education Data Set". Only three columns of the data set are used for the study. These columns are the user ID, education name, and end date of the education. There are 41213 rows in this new data set.

A new "Education Dictionary" data set is generated. All the names of education have an education ID in this dictionary data set. Then the education name column is converted to education ID column in the education data set according to this "Education

Dictionary" data set.

The end date of the education is converted from date format to a special date format that only includes month and year information. The data set is divided monthly according to this new date format. After this process, the date columns are removed from the new bipartite network data sets. All the duplicated rows are removed from the bipartite network data sets. At the end, these bipartite network data sets include only the user ID and education ID columns.

Bipartite network data sets are converted to network data sets by using the igraph library in the R programming language. So the users are connected to each other through their education IDs. The user IDs are nodes, and the education IDs are links in network files. These network files are converted to edge files, and the data preprocessing steps are ended. Finally, there are 59 edge files that were divided monthly (see Fig. 3.1).

	2017	2018	2019	2020	2021
January	E201701.edges	E201801.edges	E201901.edges	E202001.edges	E202101.edges
February	E201702.edges	E201802.edges	E201902.edges	E202002.edges	E202102.edges
March	E201703.edges	E201803.edges	E201903.edges	E202003.edges	E202103.edges
April	E201704.edges	E201804.edges	E201904.edges	E202004.edges	E202104.edges
May	E201705.edges	E201805.edges	E201905.edges	E202005.edges	E202105.edges
June	E201706.edges	E201806.edges	E201906.edges	E202006.edges	E202106.edges
July	E201707.edges	E201807.edges	E201907.edges	E202007.edges	E202107.edges
August	E201708.edges	E201808.edges	E201908.edges	E202008.edges	E202108.edges
September	E201709.edges	E201809.edges	E201909.edges	E202009.edges	E202109.edges
October	E201710.edges	E201810.edges	E201910.edges	E202010.edges	E202110.edges
November	E201711.edges	E201811.edges	E201911.edges	E202011.edges	E202111.edges
December	E201712.edges	E201812.edges	E201912.edges	E202012.edges	

FIGURE 3.1 – Edge files.

All graphs are generated from edge files by using the igraph library in the R programming language. Then they are converted from directed graphs to undirected graphs. The isolated nodes are removed from graphs, and labels are added for all nodes. Network visualizations of the first, second, third, and fourth quarters of 2017 are shown in Fig. 3.2, Fig. 3.3, Fig. 3.4 and Fig. 3.5 in order. It is possible to see the monthly changes in the education network from the visuals.

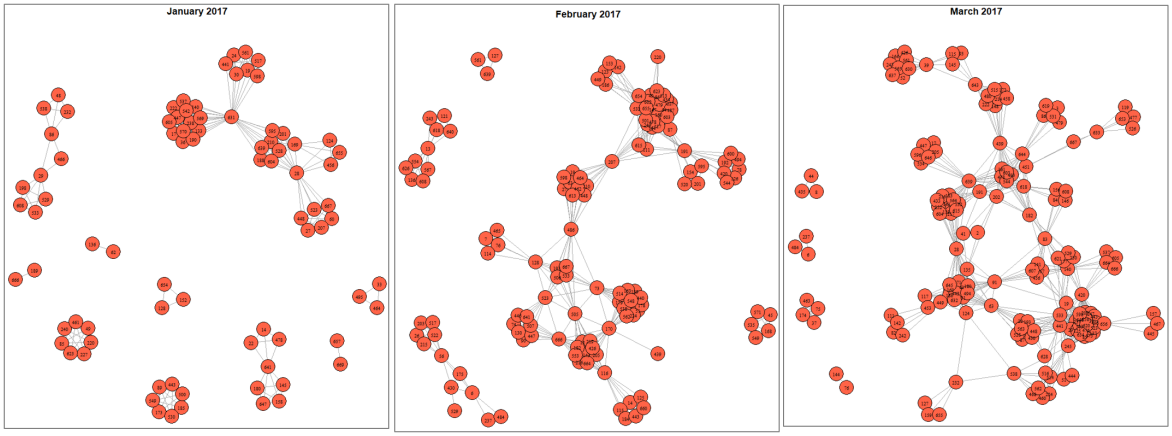


FIGURE 3.2 – Network visualization of the first quarter of 2017.

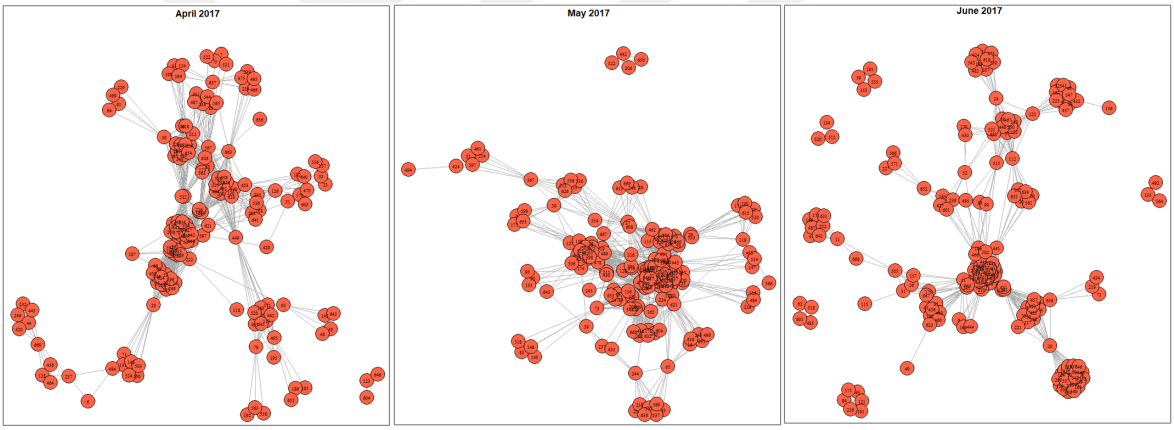


FIGURE 3.3 – Network visualization of the second quarter of 2017.

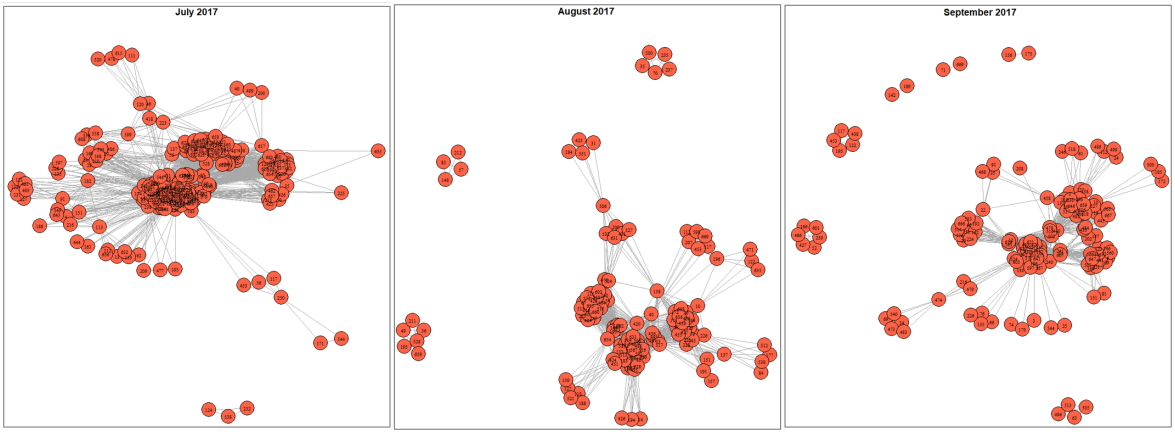


FIGURE 3.4 – Network visualization of the third quarter of 2017.

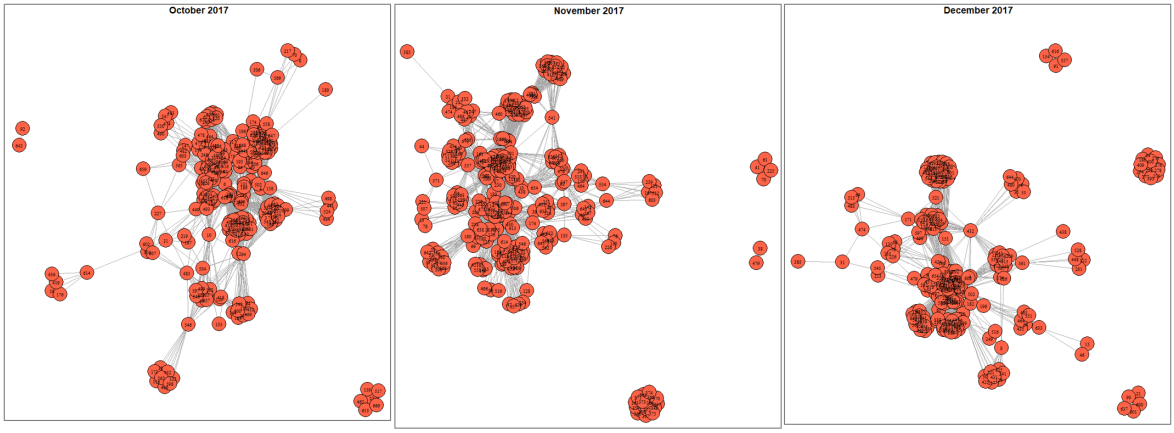


FIGURE 3.5 – Network visualization of the fourth quarter of 2017.

3.1.2 Macro Analysis

A macro-level descriptive analysis is performed for all networks, and all the results shown in the Fig 3.6, Fig 3.7, Fig 3.8, Fig 3.9, and Fig 3.10 are obtained sequentially for each year. It is possible to see all macro-level descriptive analysis results for a network and compare them with other networks. A sample visual from January 2017, which shows the degrees, layouts, and authorities in more detail, is also shown in Fig. 3.11.

	2017 Scores											
	January	February	March	April	May	June	July	August	September	October	November	December
Node Number	83	132	164	162	173	173	236	127	136	238	282	229
Link Number	280	700	939	1368	2324	1396	7212	1478	1216	3319	3655	3525
Density	0.082280341	0.080962295	0.082280341	0.082280341	0.082280341	0.082280341	0.082280341	0.082280341	0.082280341	0.082280341	0.082280341	0.082280341
Average Degree	6.746987952	10.60606061	11.45121951	16.88888889	26.86705202	16.13872832	61.11864407	23.27559055	17.88235294	27.8907563	25.92198582	30.7860262
Transitivity	0.831515152	0.834777352	0.801148461	0.749026001	0.762938501	0.881975322	0.88456179	0.786066621	0.756805808	0.741776991	0.734244906	0.756266657
Connected Component	10	5	5	2	2	6	2	4	7	3	4	4
Diameter	3	7	8	9	6	8	5	5	5	6	5	6
Radius	1	1	1	1	1	1	1	1	1	1	1	1
Average Path Length	1.898265896	3.290616114	3.322830534	3.227952881	2.400084495	3.270131207	1.947245755	2.118296782	2.332220367	2.365071534	2.49445217	2.294539565
Betweenness Centralization	0.134605357	0.290688637	0.206351826	0.183368697	0.087766321	0.170165081	0.051698655	0.102264881	0.067743307	0.078774055	0.059523628	0.15719104
Closeness Centralization	0.630109314	1.212123591	1.28934465	1.328333265	1.114117242	1.211975199	0.92585932	0.911905444	0.948013106	1.099331796	1.0952536	1.00680466
Degree Centralization	0.271378196	0.148045339	0.181280862	0.193236715	0.297284581	0.191053905	0.442048323	0.410511186	0.289760349	0.228309045	0.185331011	0.294798131
Eigenvector Centralization	0.845578369	0.83671594	0.846734229	0.745744449	0.690958548	0.782605853	0.565116055	0.688318889	0.731202043	0.750370838	0.790143883	0.736730506
Cliques	14	22	25	30	48	37	101	37	31	44	46	48

FIGURE 3.6 – Macro analysis for 2017.

	2018 Scores											
	January	February	March	April	May	June	July	August	September	October	November	December
Node Number	131	183	214	249	197	221	182	24	167	219	185	156
Link Number	750	3591	4102	4598	1980	2269	1889	77	2987	2579	2435	2140
Density	0.088079859	0.215636822	0.088079859	0.088079859	0.088079859	0.088079859	0.088079859	0.088079859	0.088079859	0.088079859	0.088079859	0.088079859
Average Degree	11.45038168	39.24590164	38.3364486	36.93172691	20.10152284	20.53393665	20.75824176	6.416666667	35.77245509	23.55251142	26.32432432	27.43589744
Transitivity	0.809888009	0.920374764	0.916403436	0.854926778	0.73413928	0.771161964	0.81568893	0.932773109	0.888444325	0.760311201	0.871550598	0.812436056
Connected Component	6	3	5	4	6	3	3	4	3	3	9	2
Diameter	4	6	5	5	5	5	5	2	6	5	4	5
Radius	1	1	1	1	1	1	1	1	1	1	1	1
Average Path Length	2.314782097	2.22652527	2.281126597	2.265851253	2.497150373	2.546047195	2.401179941	1.341880342	2.23690205	2.425926738	2.067737372	2.053387912
Betweenness Centralization	0.09276883	0.095774864	0.087258277	0.066400034	0.077288827	0.200623902	0.188322495	0.158102767	0.071858298	0.063350189	0.125760018	0.069051345
Closeness Centralization	0.764487988	0.99880171	0.935071069	0.984630274	1.081861029	1.132237361	1.080227188	0.345849802	1.017786254	1.135087181	0.874724145	0.991562625
Degree Centralization	0.188843218	0.333813727	0.336448598	0.286565617	0.218869781	0.306663924	0.29415336	0.329710145	0.308599668	0.28645637	0.351498237	0.300413565
Eigenvector Centralization	0.860465116	0.573020346	0.625696672	0.678472859	0.765786733	0.786608432	0.74848583	0.588451298	0.583467853	0.782882749	0.701005922	0.705981657
Cliques	20	77	80	76	33	43	42	11	67	42	54	41

FIGURE 3.7 – Macro analysis for 2018.

	2019 Scores											
	January	February	March	April	May	June	July	August	September	October	November	December
Node Number	139	147	205	185	216	154	130	133	225	142	171	219
Link Number	2002	2307	4821	3546	6302	2352	1324	2173	9082	1132	1728	6883
Density	0.208737358	0.214984624	0.208737358	0.208737358	0.208737358	0.208737358	0.208737358	0.208737358	0.208737358	0.208737358	0.208737358	0.208737358
Average Degree	28.8057554	31.3877551	47.03414634	38.33513514	58.35185185	30.54545455	20.36923077	32.67669173	80.72888889	15.94366197	20.21052632	62.85844749
Transitivity	0.76381782	0.730226721	0.834604803	0.883751168	0.906990218	0.867726831	0.877286483	0.878679272	0.90346041	0.664793301	0.718847936	0.773949996
Connected Component	4	2	3	4	2	6	4	3	2	5	4	2
Diameter	4	4	5	6	5	6	5	4	4	5	5	4
Radius	1	1	1	1	1	1	1	1	1	1	1	1
Average Path Length	1.993712213	1.931543758	2.076714154	2.061238736	1.935142142	2.110404268	2.387475538	1.860020435	1.729175083	2.285846332	2.251154341	1.827878995
Betweenness Centralization	0.13903322	0.112976378	0.064005228	0.139382252	0.089856465	0.08952416	0.173443243	0.110472161	0.031224544	0.112041073	0.131143564	0.048939339
Closeness Centralization	0.918262636	0.920832921	0.974773667	0.966535802	0.92169633	0.933359599	1.063085042	0.829835122	0.792696933	1.037225746	1.051980476	0.87136707
Degree Centralization	0.392711917	0.408303047	0.362577714	0.400352526	0.407665805	0.349376114	0.291711389	0.328206881	0.340496032	0.298271901	0.263467492	0.528172259
Eigenvector Centralization	0.680895598	0.64948183	0.59500226	0.633800421	0.542188827	0.653615056	0.718228061	0.571070206	0.433242838	0.733982378	0.745414736	0.564392502
Cliques	39	38	74	66	98	50	37	56	124	22	29	9

FIGURE 3.8 – Macro analysis for 2019.

	2020 Scores											
	January	February	March	April	May	June	July	August	September	October	November	December
Node Number	58	77	380	379	206	329	329	147	331	253	336	234
Link Number	243	384	64538	63837	3874	29906	35357	1828	21664	7801	31198	8512
Density	0.147005445	0.131237184	0.147005445	0.147005445	0.147005445	0.147005445	0.147005445	0.147005445	0.147005445	0.147005445	0.147005445	0.147005445
Average Degree	8.379310345	9.974025974	339.6736842	336.8707124	37.61165049	181.7993921	214.9361702	24.8707483	130.9003021	61.66798419	185.702381	72.75213675
Transitivity	0.926691103	0.888690476	0.968726991	0.975376616	0.732840353	0.930090447	0.980738991	0.826023113	0.900817477	0.898157078	0.882284274	0.897022463
Connected Component	6	6	1	1	2	1	1	4	1	1	1	3
Diameter	4	6	3	3	4	3	3	4	4	5	3	3
Radius	1	1	2	2	1	2	2	1	2	3	2	1
Average Path Length	1.901294498	2.916801729	1.106193584	1.109380017	2.037233786	1.461079398	1.365668322	2.061206989	1.679575208	1.982997679	1.465031983	1.724726967
Betweenness Centralization	0.159430008	0.300364728	0.005009292	0.004549713	0.052552891	0.011627776	0.023118497	0.094762472	0.022235346	0.057941269	0.019694028	0.035482498
Closeness Centralization	0.700039902	1.05454266	0.109300219	0.127565888	0.99593409	0.378740581	0.247854873	0.945000332	0.393071297	0.312431696	0.439553049	0.778126247
Degree Centralization	0.256503327	0.224025974	0.077378142	0.090289121	0.387260242	0.332928683	0.222755579	0.315953779	0.379089994	0.318777841	0.36506752	0.340119585
Eigenvector Centralization	0.710678449	0.765796046	0.06872388	0.067960348	0.68460135	0.289756783	0.197346331	0.682078444	0.403524244	0.536786287	0.317069157	0.495888696
Cliques	18	19	332	344	54	231	263	44	193	113	220	114

FIGURE 3.9 – Macro analysis for 2020.

	2021 Scores											
	January	February	March	April	May	June	July	August	September	October	November	December
Node Number	191	175	316	379	195	225	141	85	104	206	45	
Link Number	8181	4324	26834	63837	3170	4496	1451	515	665	8933	249	
Density	0.450868008	0.284006568	0.450868008	0.450868008	0.450868008	0.450868008	0.450868008	0.450868008	0.450868008	0.450868008	0.450868008	
Average Degree	85.66492147	49.41714286	169.835443	336.8707124	32.51282051	39.96444444	20.58156028	12.11764706	12.78846154	86.72815534	11.06666667	
Transitivity	0.935913217	0.924438144	0.964284058	0.975376616	0.828838745	0.756687838	0.890541303	0.761064087	0.642539159	0.936300735	0.987406983	
Connected Component	4	2	1	1	4	5	4	5	5	2	5	
Diameter	3	5	4	3	4	6	4	5	5	4	2	
Radius	1	1	2	2	1	1	1	1	1	1	1	
Average Path Length	1.547926295	1.978040655	1.516094033	1.109380017	2.12392652	2.021511088	2.290203327	2.270525918	2.302968037	1.646525829	1.150170648	
Betweenness Centralization	0.046873324	0.101994116	0.01545353	0.004549713	0.062812867	0.041805906	0.187599917	0.204744309	0.089765987	0.046647739	0.0421872	
Closeness Centralization	0.633640669	0.926126464	0.277223454	0.127565888	1.005117823	0.944191168	1.043314234	0.962641623	1.011701803	0.719089395	0.240133897	
Degree Centralization	0.396500413	0.422889984	0.26718907	0.090289121	0.394263812	0.303730159	0.302988855	0.331932773	0.254480956	0.337911437	0.157575758	
Eigenvector Centralization	0.369037405	0.524901435	0.278370642	0.067960348	0.685996371	0.663091627	0.722614444	0.694437591	0.739829349	0.384081999	0.604651163	
Cliques	120	83	226	344	60	61	38	18	19	126	19	

FIGURE 3.10 – Macro analysis for 2021.

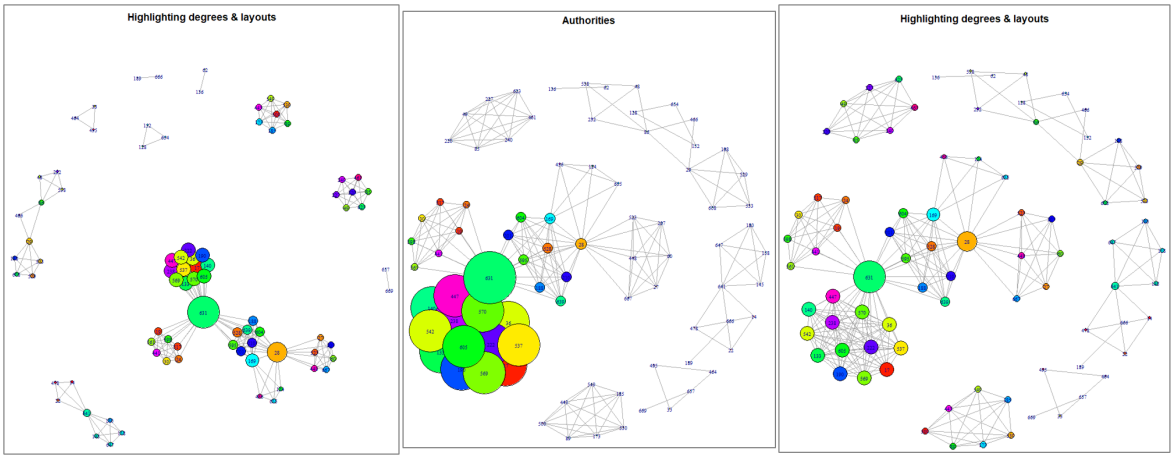


FIGURE 3.11 – Detailed network visualization for January 2017.

In Fig. 3.12, the degree centralization for January 2017 is shown as an example. The nodes with the highest degrees are 631, 28, and 169, respectively.

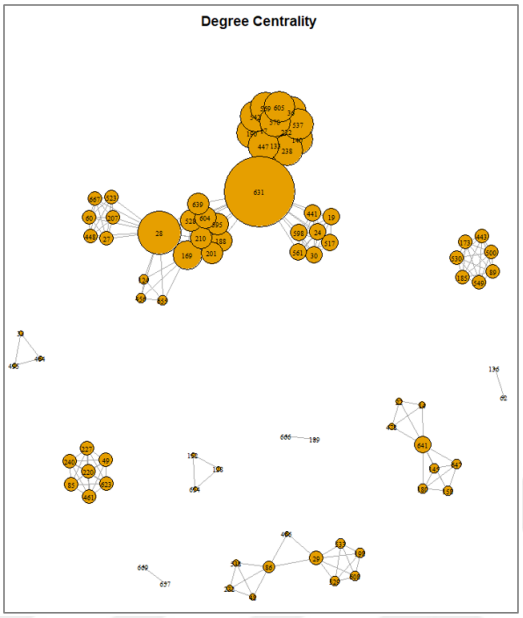


FIGURE 3.12 – Degree centralization visualization for January 2017.

In Fig. 3.13, the betweenness centralization for January 2017 is shown as an example. The nodes with the highest betweenness values are 631 and 28.

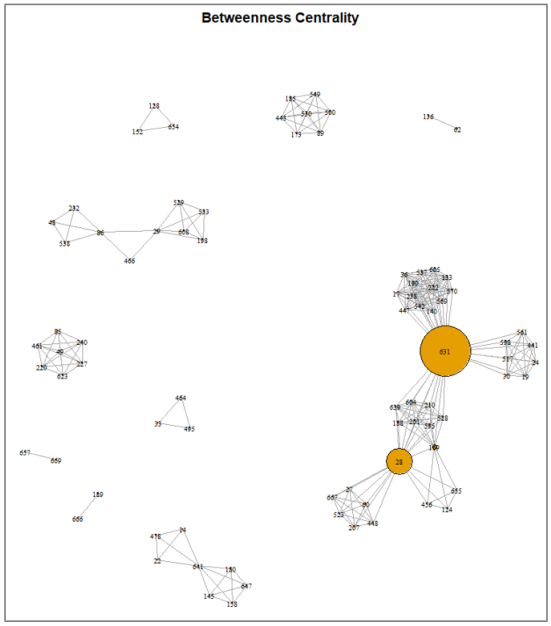


FIGURE 3.13 – Betweenness centralization visualization for January 2017.

In Fig. 3.14, the closeness centralization for January 2017 is shown as an example. The nodes with the highest closeness values are 136, 62, 657, 669, 666 and 189.

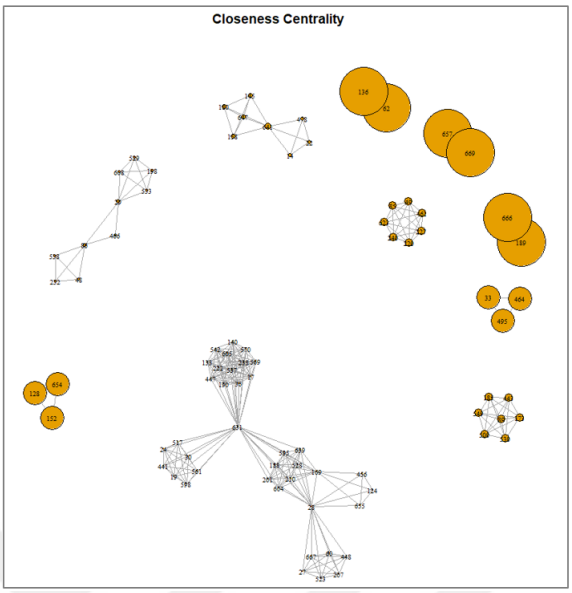


FIGURE 3.14 – Closeness centralization visualization for January 2017.

In Fig. 3.15, the eigenvector centralization for January 2017 is shown as an example. The node with the highest eigenvector value is 631.

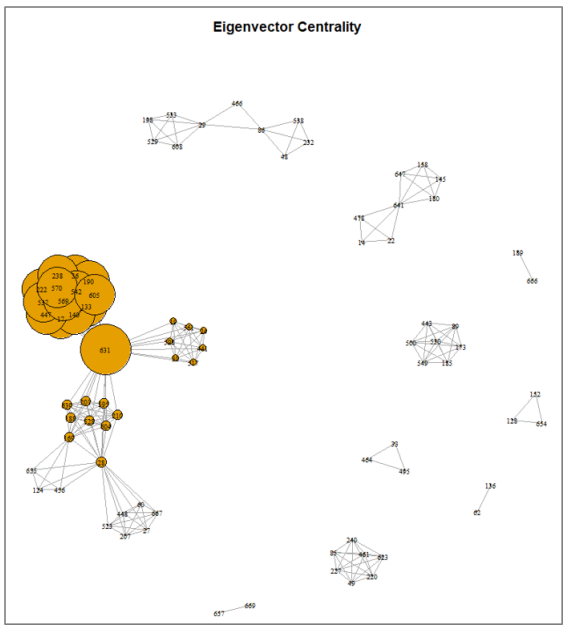
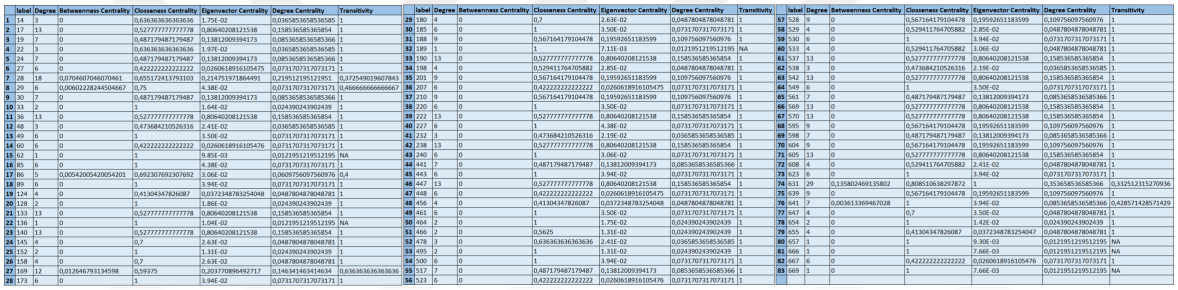


FIGURE 3.15 – Eigenvector centralization visualization for January 2017.

3.1.3 Micro Analysis

For micro-level descriptive analysis of all networks, degree, betweenness centrality, closeness centrality, eigenvector centrality, degree centrality, and transitivity scores are analyzed for each node, and all the results are saved in files. The results of the micro analysis for January 2017 are shown in Figure 3.16 as an example. It is possible to see the degree, betweenness centrality, closeness centrality, eigenvector centrality, degree centrality, and transitivity scores for each node and compare them with other nodes.



	label	Degree	Betweenness Centrality	Closeness Centrality	Eigenvector Centrality	Degree Centrality	Transitivity
1	14	3	0	0,636363636363636	1.75E-02	0,036585365853658	1
2	17	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
3	19	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
4	22	3	0	0,636363636363636	1.97E-02	0,036585365853658	1
5	24	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
6	27	6	0	0,422222222222222	0,026061891610547	0,073170731707317	1
7	28	18	0,070460704607461	0,655172413793103	0,214751971864491	0,219512195121951	0,372549019607843
8	29	6	0,006022282445046	0,75	4.38E-02	0,073170731707317	0,466666666666667
9	30	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
10	33	2	0	1	1.64E-02	0,024390243902439	1
11	36	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
12	48	3	0	0,473684210526316	2.41E-02	0,036585365853658	1
13	49	6	0	1	3.50E-02	0,073170731707317	1
14	60	6	0	0,422222222222222	0,026061891610547	0,073170731707317	1
15	62	1	0	1	9.85E-03	0,012195121951219	NA
16	85	6	0	1	4.38E-02	0,073170731707317	1
17	86	5	0,005420054200542	0,692307692307692	3.06E-02	0,060975609756098	0,4
18	89	6	0	1	3.94E-02	0,073170731707317	1
19	124	4	0	0,413043478260870	0,037234878325404	0,048780487804878	1
20	128	2	0	1	1.86E-02	0,024390243902439	1
21	133	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
22	136	1	0	1	1.04E-02	0,012195121951219	NA
23	140	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
24	145	6	0	0,7	2.63E-02	0,048780487804878	1
25	152	2	0	1	1.31E-02	0,024390243902439	1
26	158	4	0	0,7	2.63E-02	0,048780487804878	1
27	169	12	0,012646793134598	0,59375	0,203770896492717	0,146341463414634	0,636363636363636
28	173	6	0	1	3.94E-02	0,073170731707317	1
29	180	4	0	0,7	2.63E-02	0,048780487804878	1
30	185	6	0	1	3.50E-02	0,073170731707317	1
31	188	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
32	189	1	0	1	7.11E-03	0,012195121951219 5	NA
33	190	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
34	198	4	0	0,529411764705882	2.85E-02	0,048780487804878	1
35	201	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
36	207	6	0	0,422222222222222	0,026061891610547 5	0,073170731707317	1
37	210	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
38	220	6	0	1	3.50E-02	0,073170731707317	1
39	222	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
40	227	6	0	1	4.38E-02	0,073170731707317	1
41	232	3	0	0,473684210526316	2.19E-02	0,036585365853658 5	1
42	238	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
43	240	6	0	1	3.06E-02	0,073170731707317	1
44	441	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
45	443	6	0	1	3.94E-02	0,073170731707317	1
46	447	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
47	448	6	0	0,422222222222222	0,026061891610547 5	0,073170731707317	1
48	456	4	0	0,413043478260870	0,037234878325404 8	0,048780487804878	1
49	461	6	0	1	3.50E-02	0,073170731707317	1
50	464	2	0	1	1.75E-02	0,024390243902439	1
51	466	2	0	0,5625	1.31E-02	0,024390243902439	1
52	478	3	0	0,636363636363636	2.41E-02	0,036585365853658 5	1
53	495	2	0	1	1.31E-02	0,024390243902439	1
54	500	6	0	1	3.94E-02	0,073170731707317	1
55	517	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
56	523	6	0	0,422222222222222	0,026061891610547 6	0,073170731707317	1
57	528	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
58	529	4	0	0,529411764705882	2.85E-02	0,048780487804878	1
59	530	6	0	1	3.94E-02	0,073170731707317	1
60	533	4	0	0,529411764705882	3.06E-02	0,048780487804878	1
61	537	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
62	538	3	0	0,473684210526316	2.19E-02	0,036585365853658 5	1
63	542	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
64	549	6	0	1	3.50E-02	0,073170731707317	1
65	561	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
66	569	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
67	570	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
68	595	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
69	598	7	0	0,487179487179487	0,138120099394173	0,085365853658366	1
70	604	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
71	605	13	0	0,527777777777778	0,80640208121538	0,158536585365854	1
72	608	4	0	0,529411764705882	2.41E-02	0,048780487804878	1
73	623	6	0	1	3.94E-02	0,073170731707317	1
74	631	29	0,135802469135802	0,808510638297872	1	0,353658536585366	0,332512315270936
75	639	9	0	0,567164179104478	0,195926511835 99	0,109756097560976	1
76	641	7	0,003613369467028	1	3.94E-02	0,085365853658366	0,428571428571429
77	647	4	0	0,7	3.50E-02	0,048780487804878	1
78	654	2	0	1	1.42E-02	0,024390243902439	1
79	655	4	0	0,413043478260870	0,037234878325404 7	0,048780487804878	1
80	657	1	0	1	9.30E-03	0,012195121951219 5	NA
81	666	1	0	1	7.66E-03	0,012195121951219 5	NA
82	667	6	0	0,422222222222222	0,026061891610547 6	0,073170731707317	1
83	669	1	0	1	7.66E-03	0,012195121951219 5	NA

FIGURE 3.16 – Micro analysis for January 2017.

Degree distribution in network science refers to the probability distribution of the degrees of nodes in a network. In a network, the degree of a node represents the number of connections or edges it has with other nodes. In many real-world networks, such as social networks, the degree distribution follows a power law. This means that there are a few nodes with a very high degree and a large number of nodes with a low degree. The data used in this study also shows power law distribution in general because it is a real-world network. Figure 3.17 shows the degree distribution for the January 2017 network. The degree distribution gives an idea about the structure of the network. For the January 2017 network, there are only a few nodes (employees) whose degree is higher than 15. In general, there are nodes with low degrees. The network in January 2017 is shared only for illustrative purposes ; similar situations are observed in other networks as well. Some examples from other networks have been included in the appendices section of this study. Similar effects of power law distribution can also be observed in other networks.

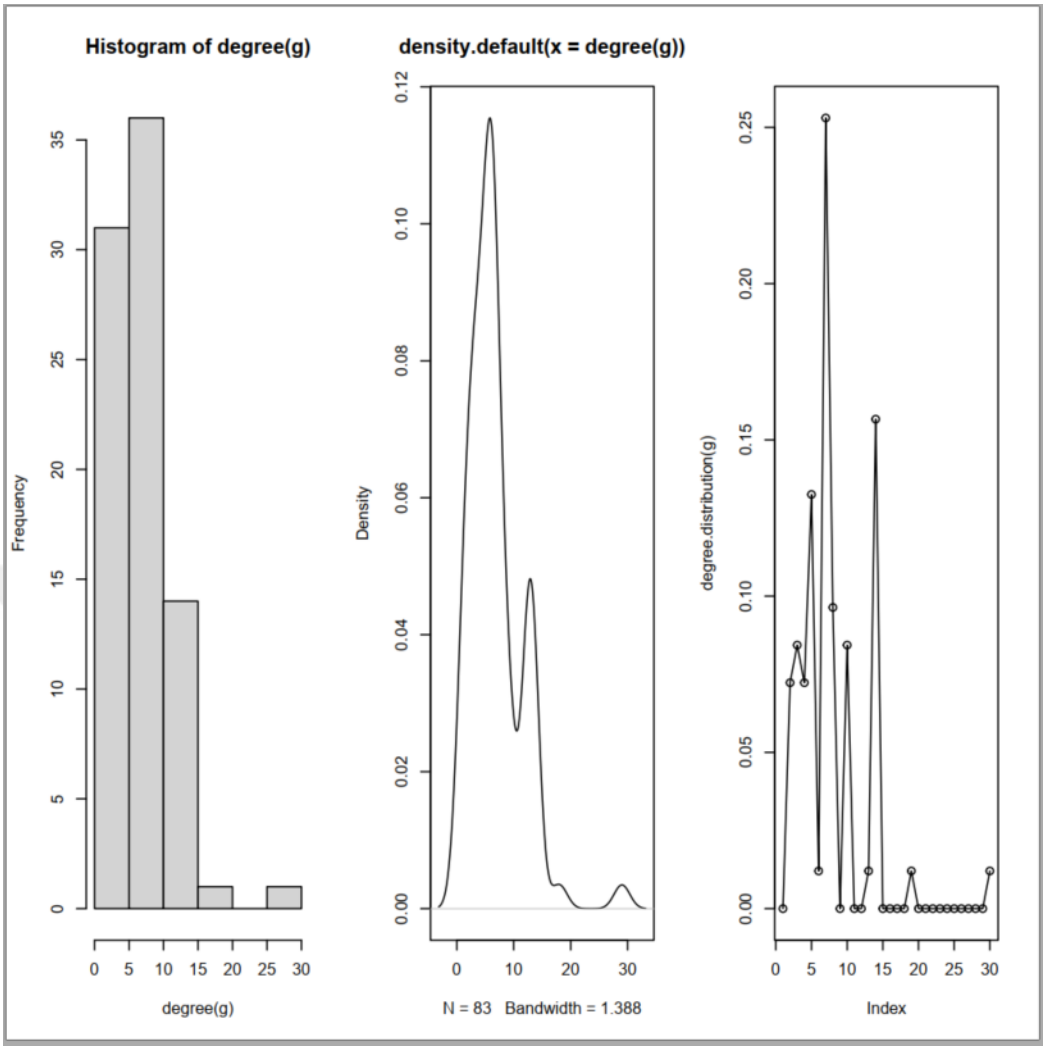


FIGURE 3.17 – Degree distribution for January 2017.

3.1.4 Meso Analysis

In this part of the study, meso-level descriptive analysis is performed for all networks, and the results for the years 2017, 2018, and 2019 are shown in Fig 3.18, Fig 3.19 and Fig 3.20, respectively. Furthermore, example visuals for January 2017 are provided to reinforce this information. Visuals are created based on the community membership of each employee. Each employee is located in the same community as the other employees they are related to.

The edge betweenness, fast greedy, infomap, label propagation, leading eigenvector, louvain, and walktrap scores for each month of the year 2017 are shown in Fig. 3.18.

	2017 Modularities											
	January	February	March	April	May	June	July	August	September	October	November	December
Edge Betweenness	0.725184949	0.688738776	0.660385768	0.495083423	0.352980369	0.595548528	0.284239304	0.395615166	0.471347291	0.563753789	0.600849688	0.458640229
Fast Greedy	0.727633929	0.670964286	0.620575556	0.489891655	0.376281435	0.583334958	0.315283778	0.424722955	0.44973273	0.533664855	0.597931698	0.46381007
Infomap	0.725184949	0.683710204	0.660851335	0.492178683	0.33493326	0.599021406	0.291764796	0.411411116	0.479278142	0.572193832	0.594674798	0.440063256
Label Propagation	0.715790816	0.675566327	0.646166418	0.31672085	0.28322955	0.591699011	0.278627095	0.40649119	0.476554046	0.545618319	0.584380035	0.433038821
Leading Eigenvector	0.727404337	0.669111224	0.641549878	0.485675347	0.394665427	0.59990425	0.311578893	0.432000646	0.47171925	0.567071217	0.599409538	0.497432483
Louvain	0.730848214	0.697431633	0.674440101	0.509061816	0.407196033	0.605847345	0.325961438	0.443476354	0.488560896	0.574543284	0.616313503	0.500991781
Walktrap	0.725184917	0.674044898	0.662143694	0.490815016	0.339241648	0.59525399	0.295630286	0.407004583	0.403402482	0.53137037	0.589326429	0.471602113

FIGURE 3.18 – Meso analysis for 2017.

The edge betweenness, fast greedy, infomap, label propagation, leading eigenvector, louvain, and walktrap scores for each month of the year 2018 are shown in Fig. 3.19.

	2018 Modularities											
	January	February	March	April	May	June	July	August	September	October	November	December
Edge Betweenness	0.710584889	0.186506831	0.224960306	0.373053601	0.517412254	0.531912389	0.522365	0.372069489	0.255027113	0.506601099	0.444786207	0.560694275
Fast Greedy	0.696738667	0.211732455	0.258016538	0.346205047	0.497513264	0.523416481	0.514086457	0.393320965	0.254455055	0.477485104	0.379617235	0.470453206
Infomap	0.704042667	0.189653642	0.262997251	0.381506342	0.529553872	0.544748513	0.522233005	0.372069489	0.25533057	0.518067606	0.445393791	0.560949865
Label Propagation	0.642872	0.164246169	0.182407215	0.385018847	0.4760754	0.534254591	0.50928602	0.372069489	0.13035457	0.510941492	0.386124409	0.562858765
Leading Eigenvector	0.696486222	0.209886818	0.28180225	0.397793635	0.506001046	0.541317907	0.512854225	0.393320965	0.269568196	0.497611198	0.456573751	0.560922024
Louvain	0.715596444	0.222661375	0.285763167	0.406515584	0.541839353	0.55792727	0.540567959	0.393320965	0.282411597	0.533795423	0.458998351	0.568047209
Walktrap	0.709144	0.186152128	0.253628345	0.384965043	0.525173197	0.512735412	0.521485595	0.372069489	0.254075438	0.514411521	0.444212186	0.566957485

FIGURE 3.19 – Meso analysis for 2018.

The edge betweenness, fast greedy, infomap, label propagation, leading eigenvector, louvain, and walktrap scores for each month of the year 2019 are shown in Fig. 3.20.

	2019 Modularities											
	January	February	March	April	May	June	July	August	September	October	November	December
Edge Betweenness	0.465124661	0.357332492	0.358794092	0.420785604	0.249389944	0.445736405	0.527431921	0.249982158	0.080694419	0.496284212	0.562418285	0.165725822
Fast Greedy	0.444765898	0.363786907	0.358443972	0.375192141	0.27409683	0.428494511	0.502400774	0.294353642	0.125889962	0.492460341	0.484524197	0.20078242
Infomap	0.432103236	0.359947176	0.362075775	0.426005335	0.256501565	0.449199227	0.530165558	0.249301504	0.074594463	0.507462791	0.569686743	0.224278997
Label Propagation	0.397963176	0.359306	0.257285559	0.381147072	0.255140978	0.450600914	0.324350704	0.203918886	0.000220192	0.400081472	0.492898991	0.097492946
Leading Eigenvector	0.450303817	0.410784802	0.39478878	0.431366388	0.284860717	0.447678864	0.542042789	0.285017839	0.120747328	0.498787677	0.529746476	0.276421162
Louvain	0.483779208	0.42683469	0.397810803	0.440387698	0.292566532	0.461705771	0.550020936	0.30382816	0.141258659	0.523060283	0.581697022	0.281541657
Walktrap	0.462356824	0.374059913	0.361373341	0.428254479	0.254508752	0.455123408	0.540855094	0.259350257	0.095935768	0.404680652	0.554136928	0.241643578

FIGURE 3.20 – Meso analysis for 2019.

The community detection visualization generated based on the edge betweenness scores for January 2017 is shown in Figure 3.21 as an example of cluster edge betweenness. Each node is located within its corresponding community and colored accordingly.

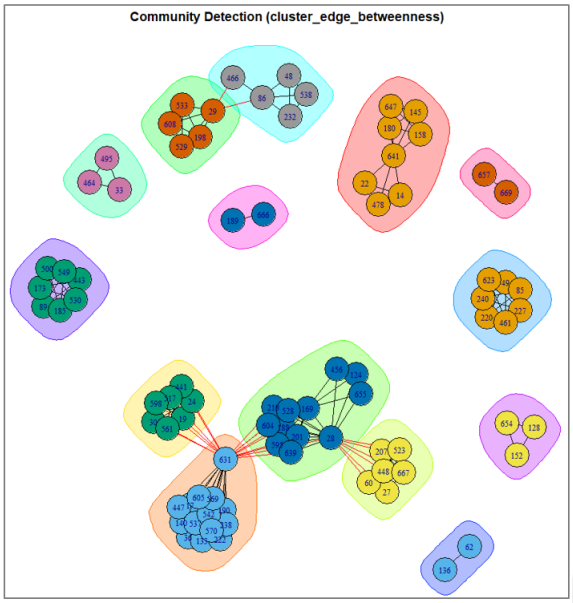


FIGURE 3.21 – Community detection for January 2017 (Cluster Edge Betweenness).

The community detection visualization generated based on the fast greedy scores for January 2017 is shown in Figure 3.22 as an example of cluster fast greedy. Each node is located within its corresponding community and colored accordingly.

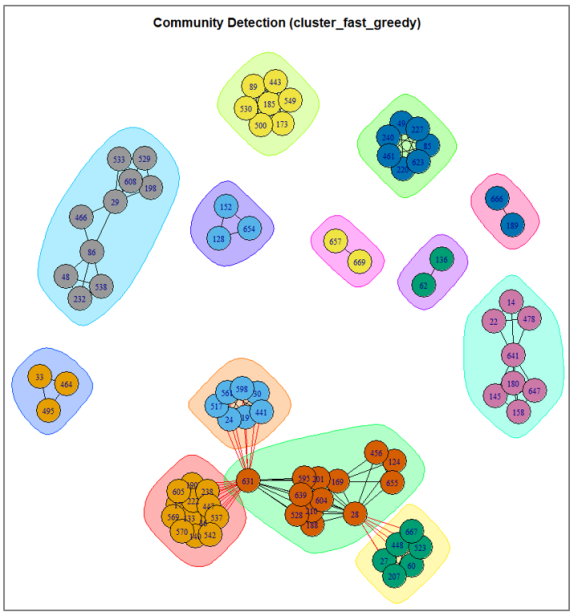


FIGURE 3.22 – Community detection for January 2017 (Cluster Fast Greedy).

The community detection visualization generated based on the infomap scores for January 2017 is shown in Figure 3.23 as an example of cluster infomap. Each node is located within its corresponding community and colored accordingly.

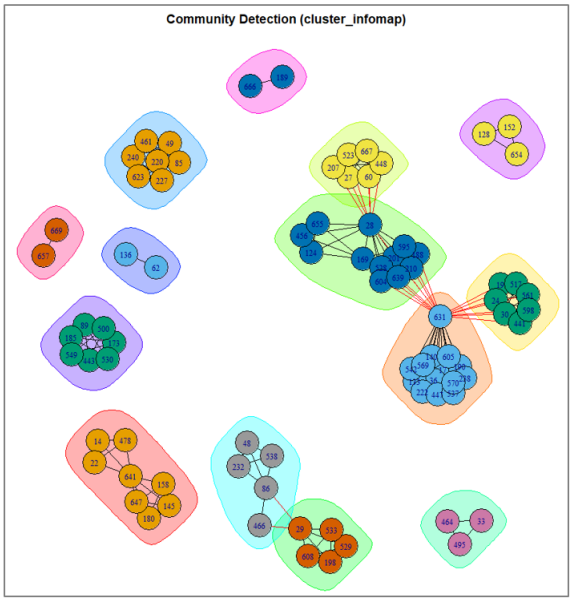


FIGURE 3.23 – Community detection for January 2017 (Cluster Infomap).

The community detection visualization generated based on the label propagation scores for January 2017 is shown in Figure 3.24 as an example of cluster label propagation. Each node is located within its corresponding community and colored accordingly.

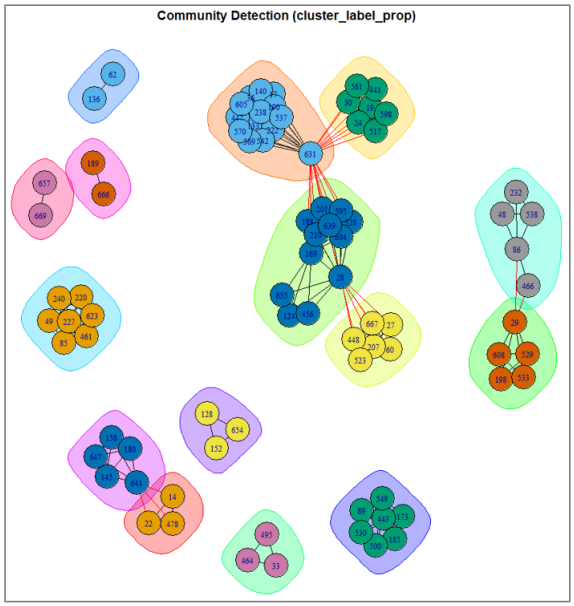


FIGURE 3.24 – Community detection for January 2017 (Cluster Label Propagation).

The community detection visualization generated based on the leading eigenvector scores for January 2017 is shown in Figure 3.25 as an example of cluster leading eigenvector. Each node is located within its corresponding community and colored accordingly.

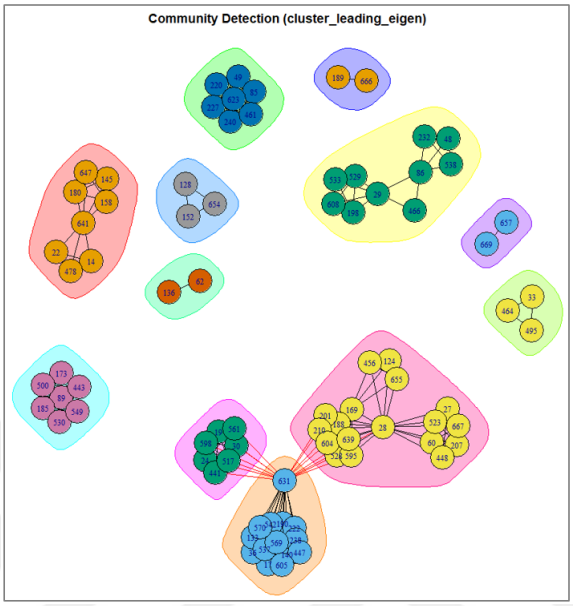


FIGURE 3.25 – Community detection for January 2017 (Cluster Leading Eigenvector).

The community detection visualization generated based on the louvain scores for January 2017 is shown in Figure 3.26 as an example of cluster louvain. Each node is located within its corresponding community and colored accordingly.

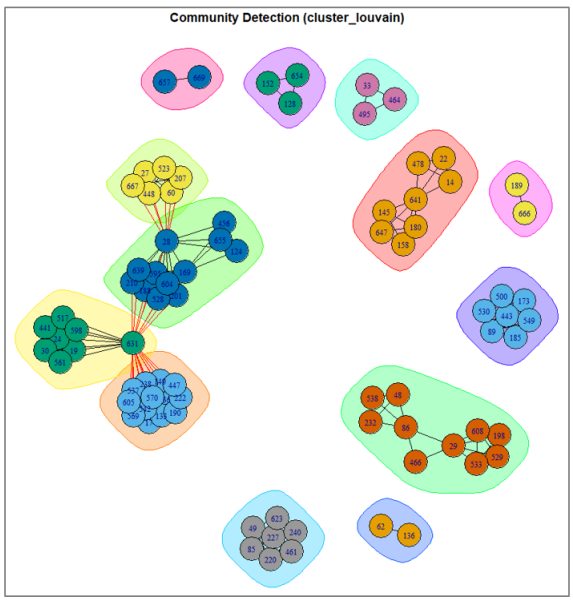


FIGURE 3.26 – Community detection for January 2017 (Cluster Louvain).

The community detection visualization generated based on the walktrap scores for January 2017 is shown in Figure 3.27 as an example of cluster walktrap. Each node is located within its corresponding community and colored accordingly.

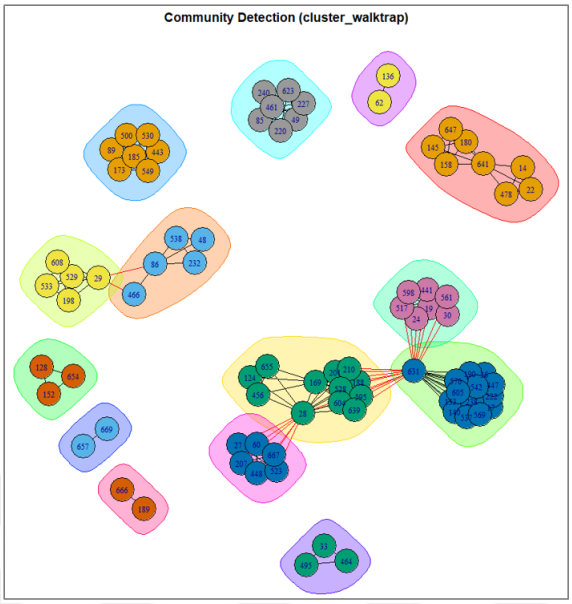


FIGURE 3.27 – Community detection for January 2017 (Cluster Walktrap).

After the community detection process, heatmaps are created to visualize the community memberships for the respective network. In Figure 3.28, the community membership heatmap is represented for January 2017 network. It is possible to get information about the modular structure and organization of the network by using heatmaps. It helps to understand the underlying structure and dynamics of complex networks. It can show distinct communities, overlapping communities, or hierarchical relationships among communities. Community membership heatmaps are used for network analysis, social network analysis, and community detection algorithms.

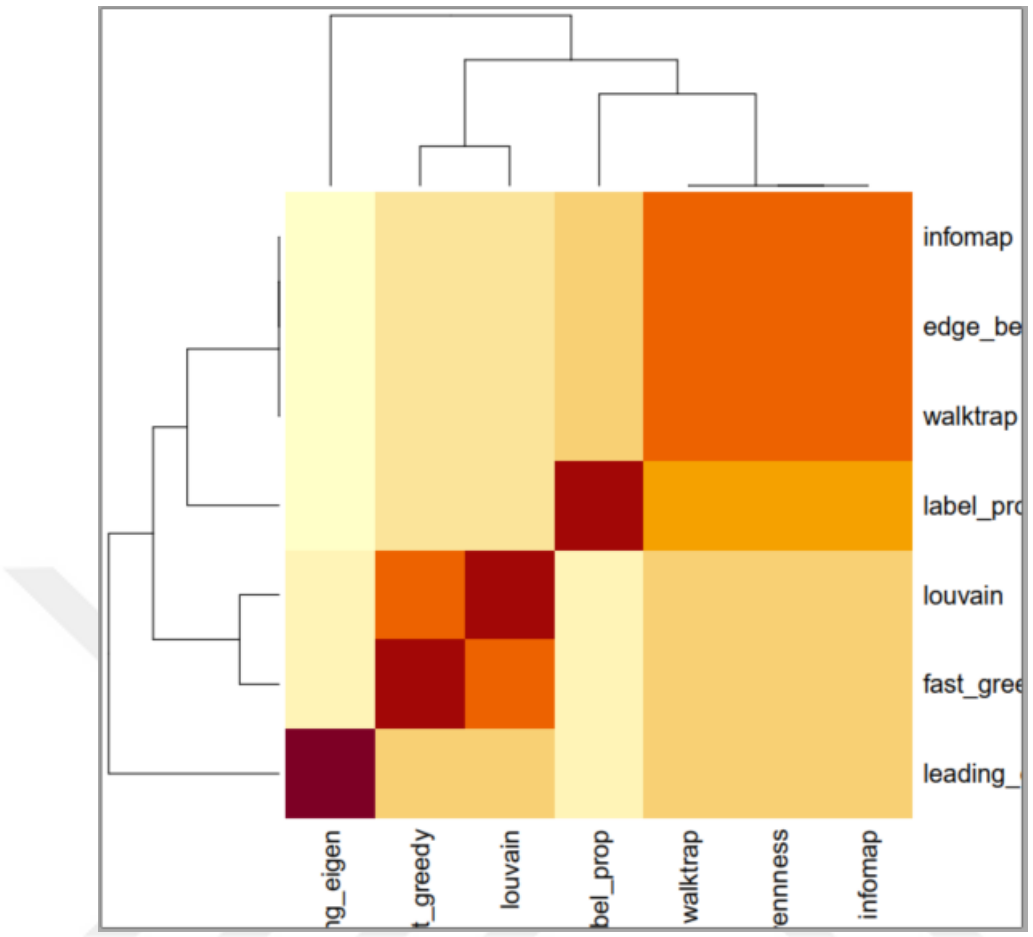


FIGURE 3.28 – Community Memberships Heatmap for January 2017.

3.1.5 Time Series Analysis

In this part of the study, the changes for variables in macro files are examined according to time, and all time series plots are saved.

3.1.5.1 ARIMA

After time series analysis, the autoregressive integrated moving average (ARIMA) model is applied to variables in macro files, and all the results and plots are saved.

(H. and A., 2018) was used as a resource to learn about the ARIMA model and how to use it in the R programming language. Different ways were tried, but in the end it was decided that the ARIMA model was not a suitable model for this study. Due to the effects of the pandemic, consistent information cannot be obtained from the data with ARIMA.

When looking at all the values, the impact of the COVID-19 pandemic can be observed. The COVID-19 pandemic started to have an impact on work life in March 2020. Increases and disruptions in the regular flow are noticeable on the relevant dates. This situation makes community detection in dynamic networks difficult. Dates such as August 2018, where there is a decrease despite a high number of nodes, indicate periods of collective leave. The time series plot and ARIMA plot for each variable are presented together in the following visuals, respectively. The purple areas in the ARIMA plots represent forecasting based on the time series analysis. However, due to the irregularities in the graph, it can generally be said that the forecasts were not successful.

Fig. 3.29 and Fig. 3.30 show results for node number. Fig. 3.31 and Fig. 3.32 show results for link number. Fig. 3.33 and Fig. 3.34 show results for density. Fig. 3.35 and Fig. 3.36 show results for average degree. Fig. 3.37 and Fig. 3.38 show results for transitivity. Fig. 3.39 and Fig. 3.40 show results for connected component. Fig. 3.41 and Fig. 3.42 show results for diameter. Fig. 3.43 and Fig. 3.44 show results for radius. Fig. 3.45 and Fig. 3.46 show results for average path length. Fig. 3.47 and Fig. 3.48 show results for betweenness centralization. Fig. 3.49 and Fig. 3.50 show results for closeness centralization. Fig. 3.51 and Fig. 3.52 show results for degree centralization. Fig. 3.53 and Fig. 3.54 show results for eigenvector centralization. Fig. 3.55 and Fig. 3.56 show results for cliques.

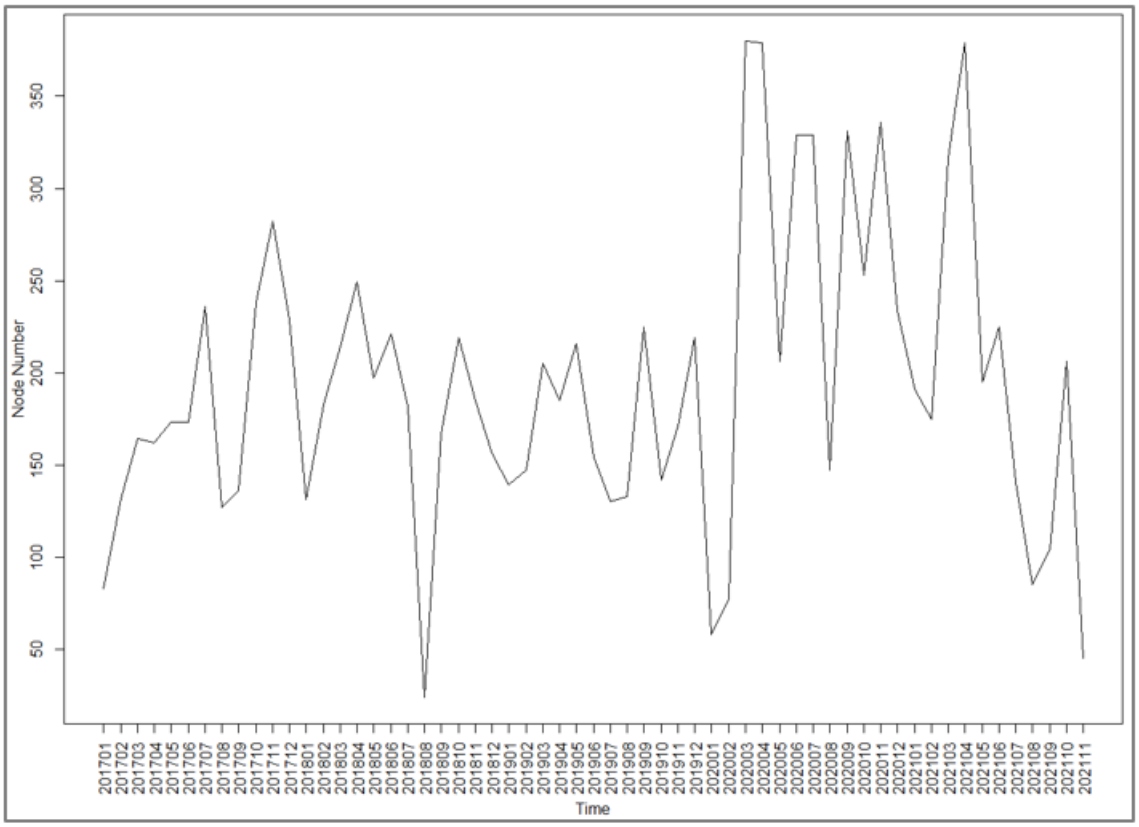


FIGURE 3.29 – Time series analysis for node number.

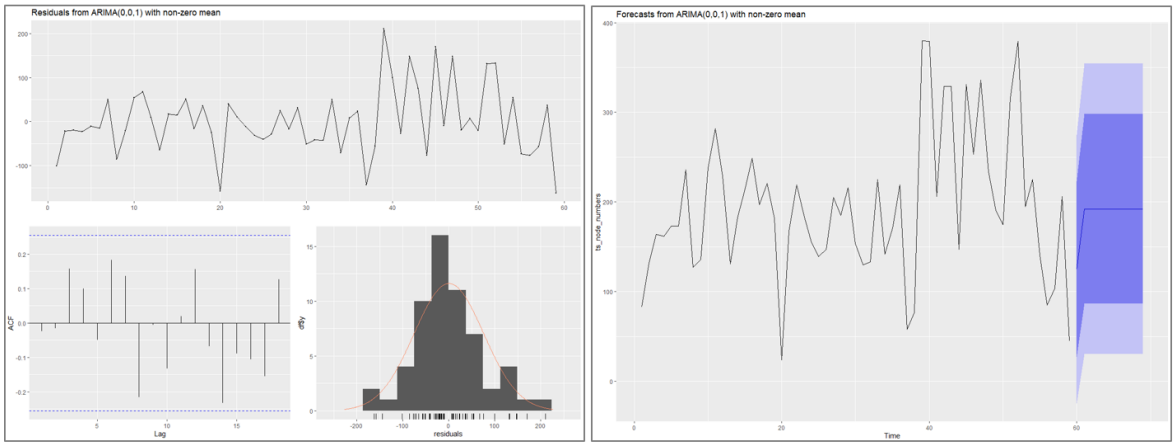


FIGURE 3.30 – ARIMA for node number.

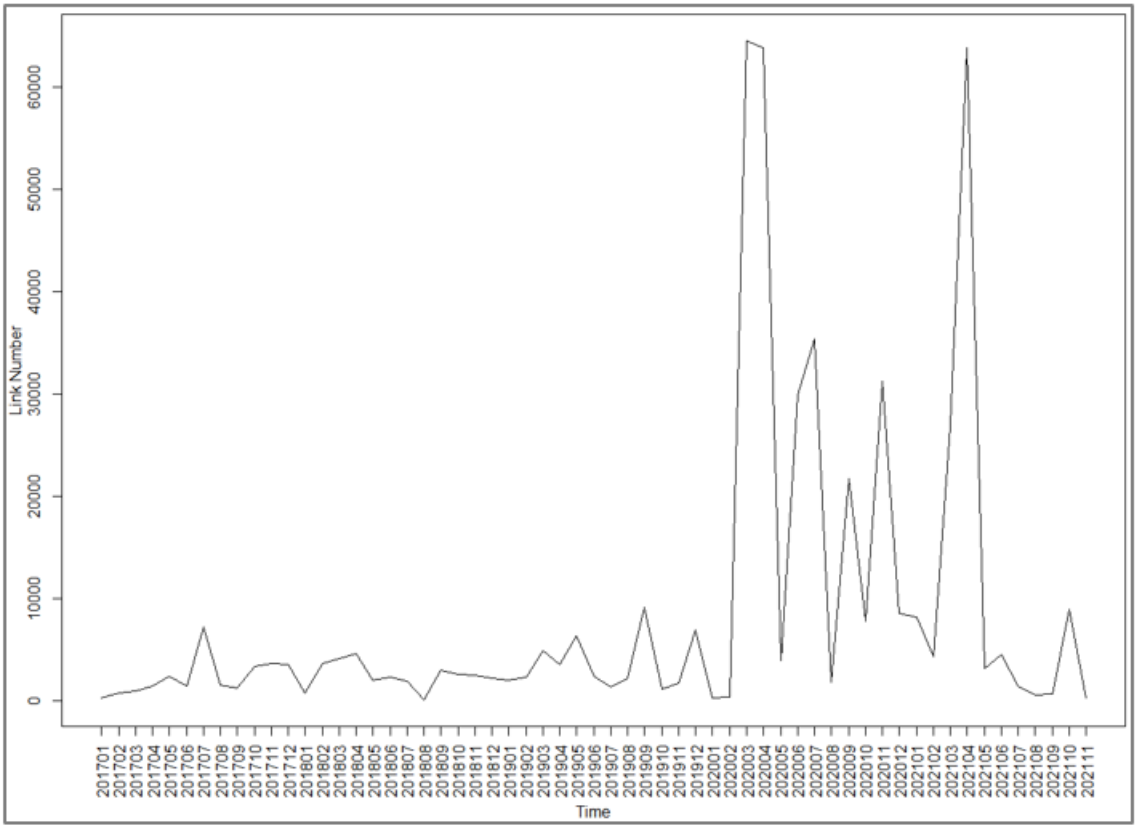


FIGURE 3.31 – Time series analysis for link number.

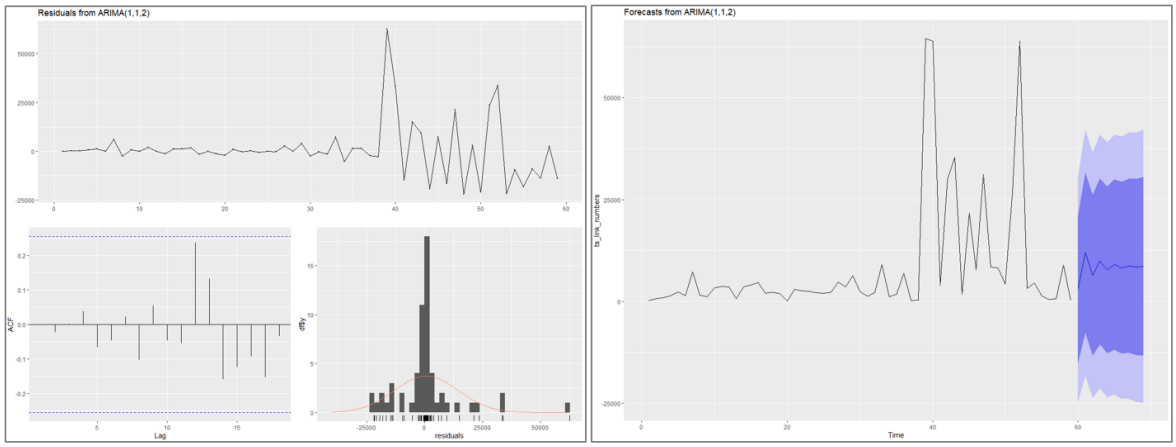


FIGURE 3.32 – ARIMA for link number.

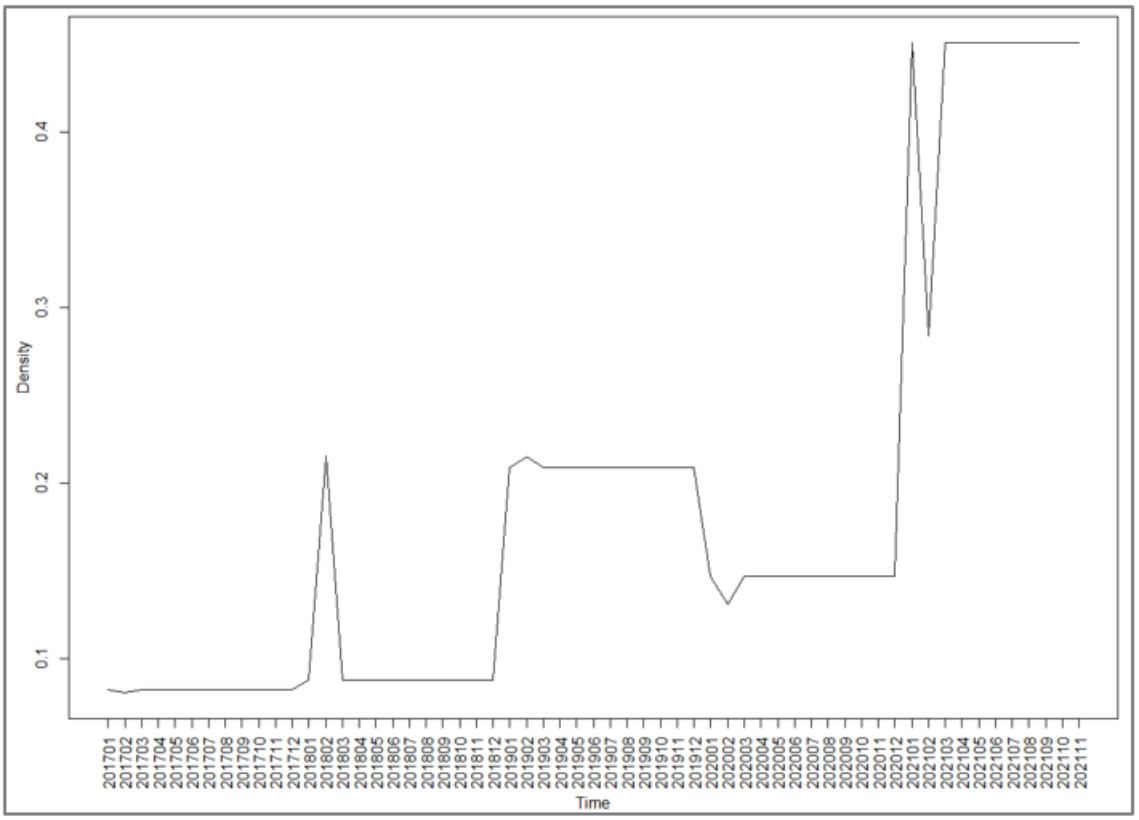


FIGURE 3.33 – Time series analysis for density.

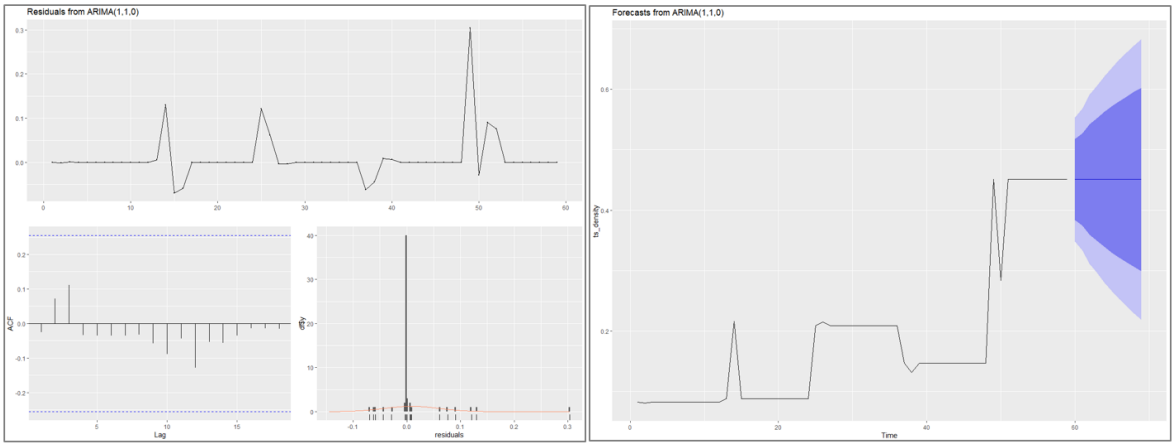


FIGURE 3.34 – ARIMA for density.

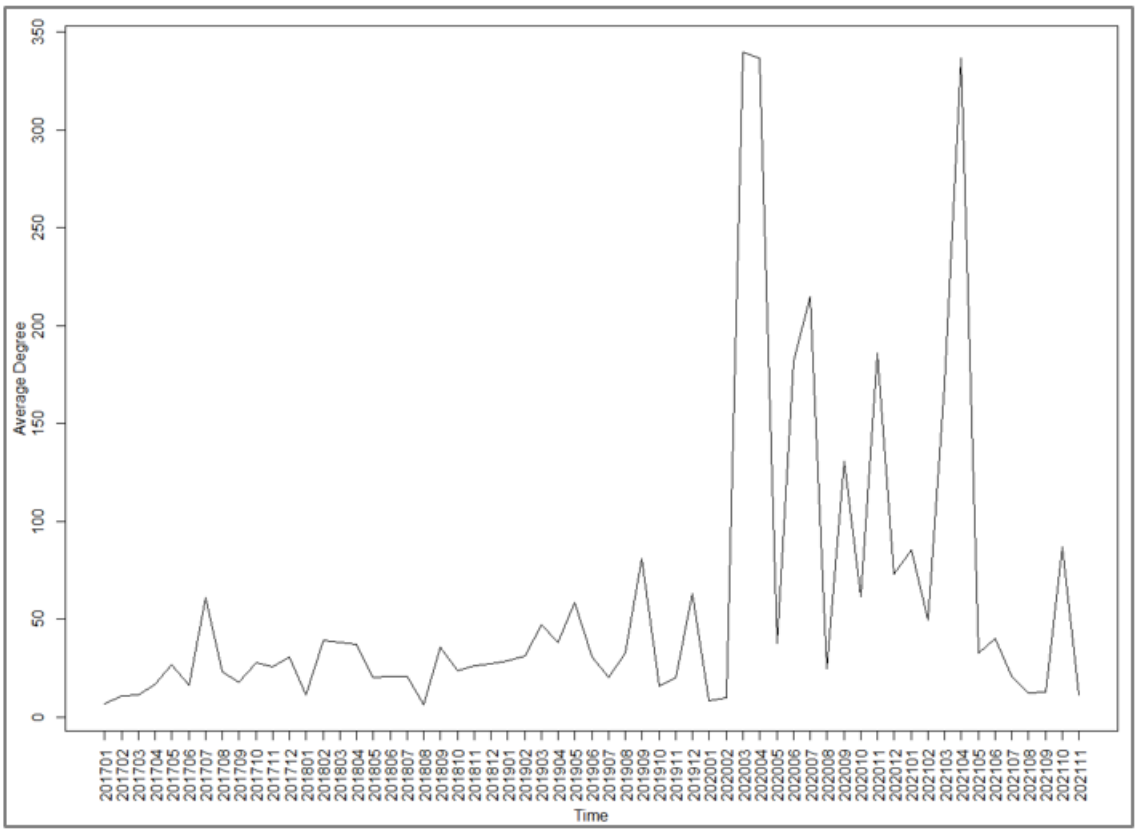


FIGURE 3.35 – Time series analysis for average degree.

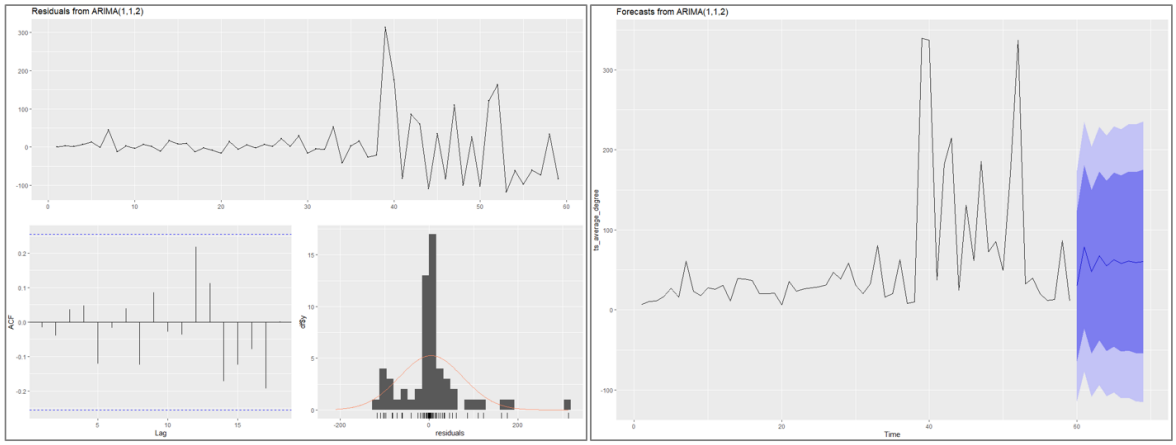


FIGURE 3.36 – ARIMA for average degree.

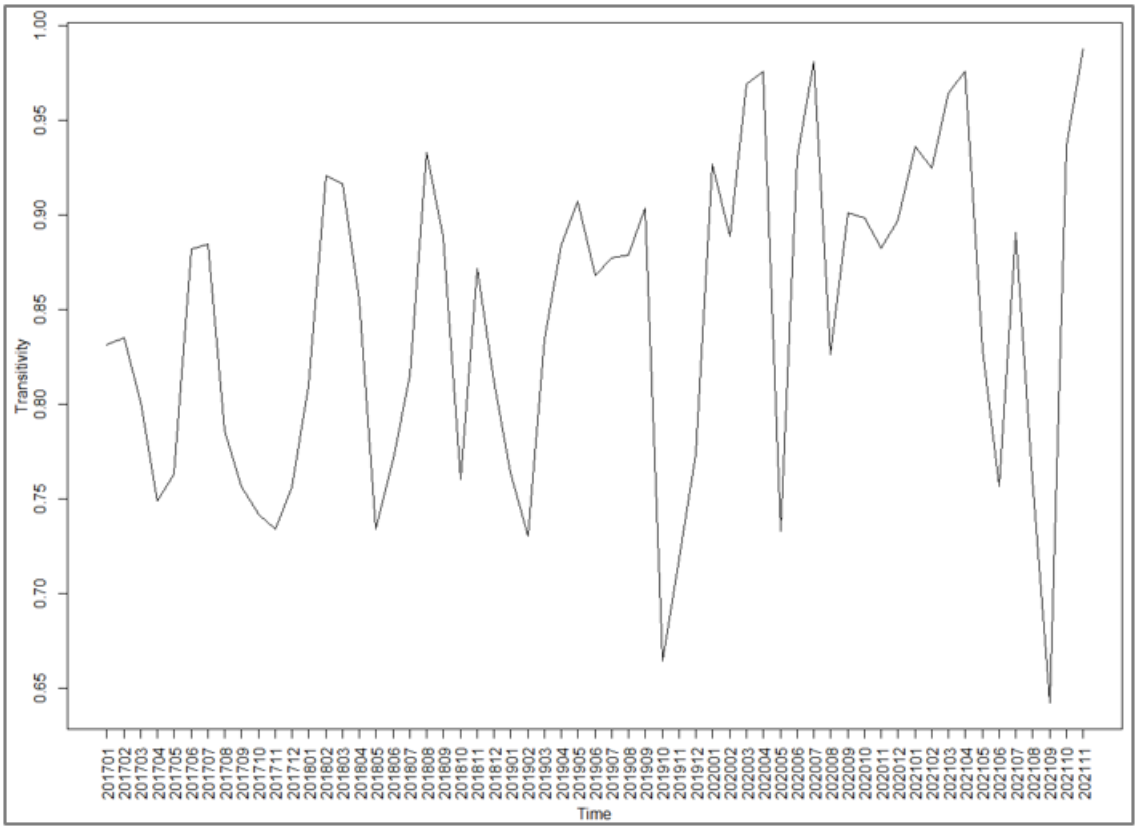


FIGURE 3.37 – Time series analysis for transitivity.

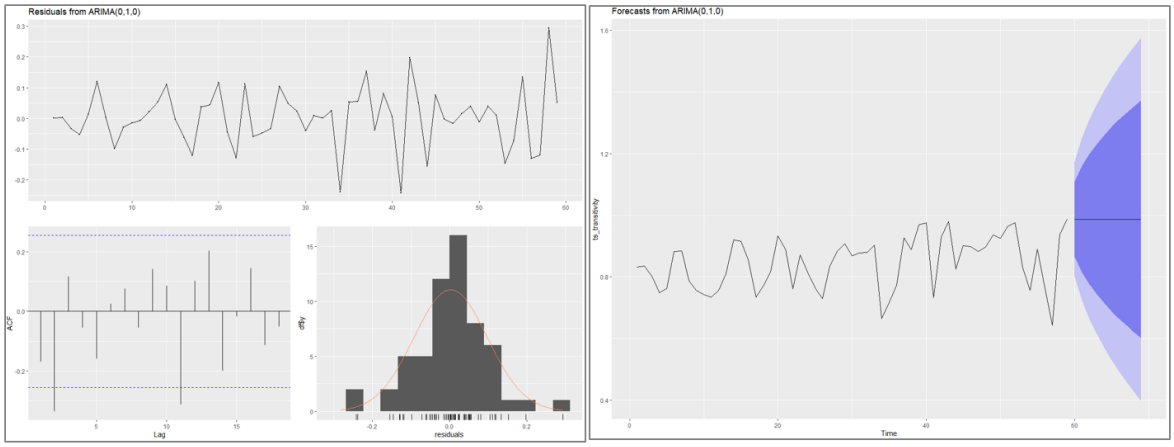


FIGURE 3.38 – ARIMA for transitivity.

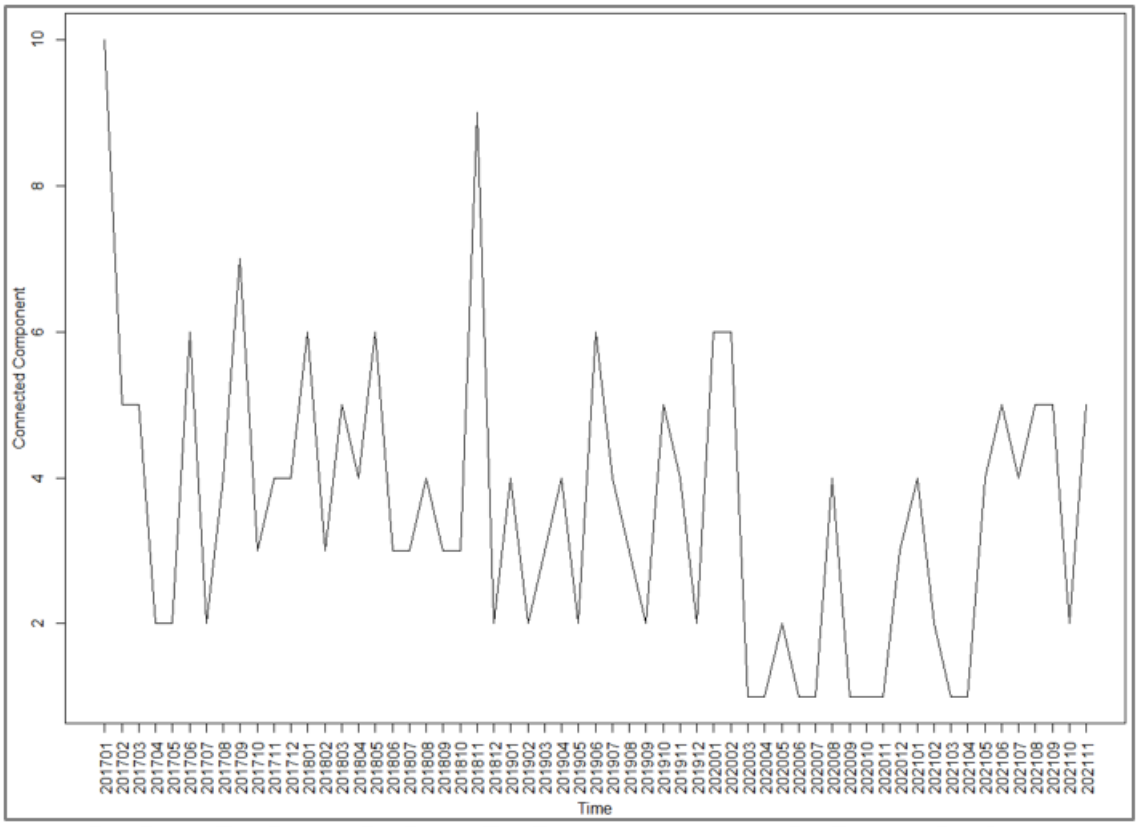


FIGURE 3.39 – Time series analysis for connected component.

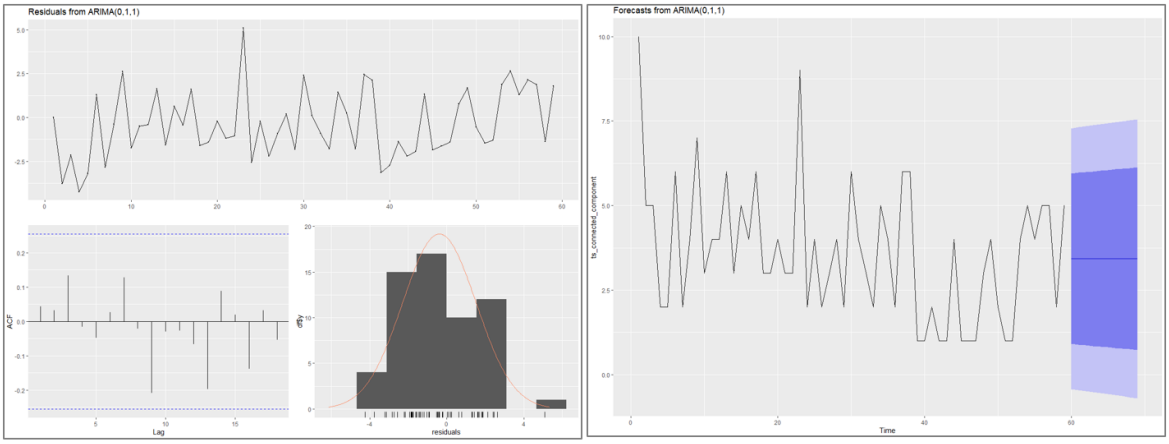


FIGURE 3.40 – ARIMA for connected component.

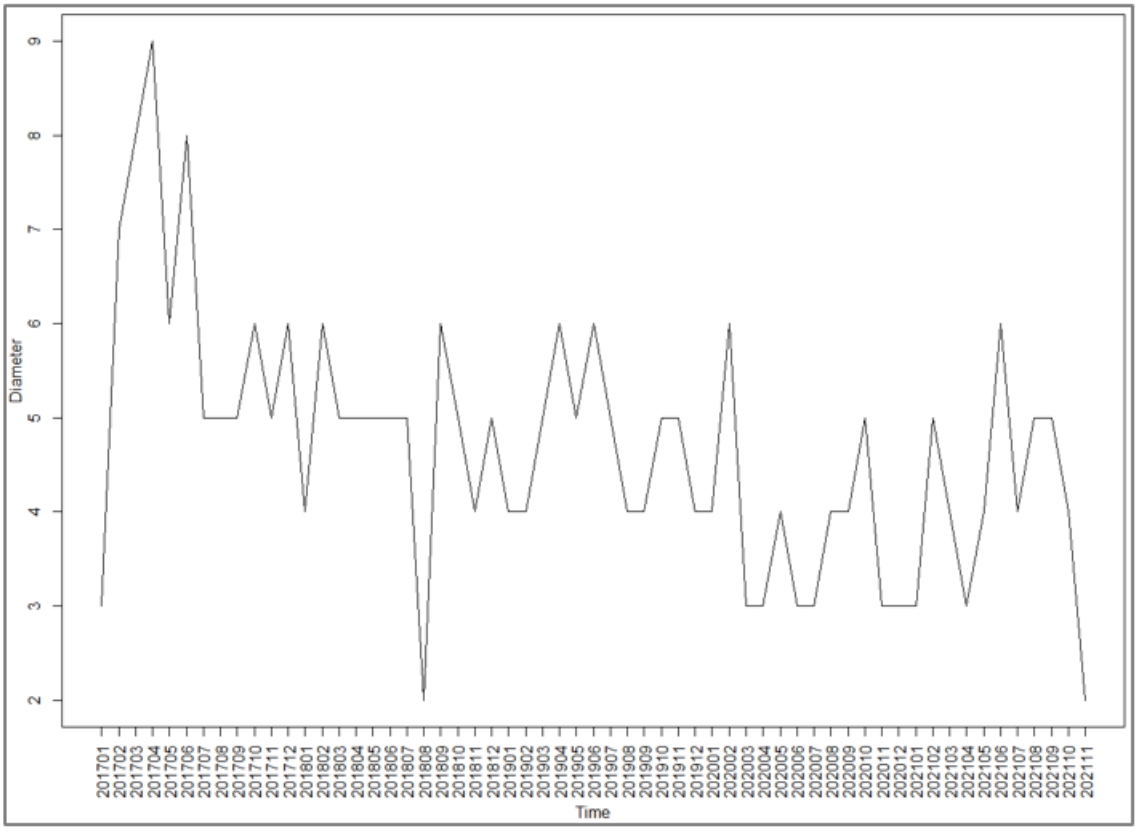


FIGURE 3.41 – Time series analysis for diameter.

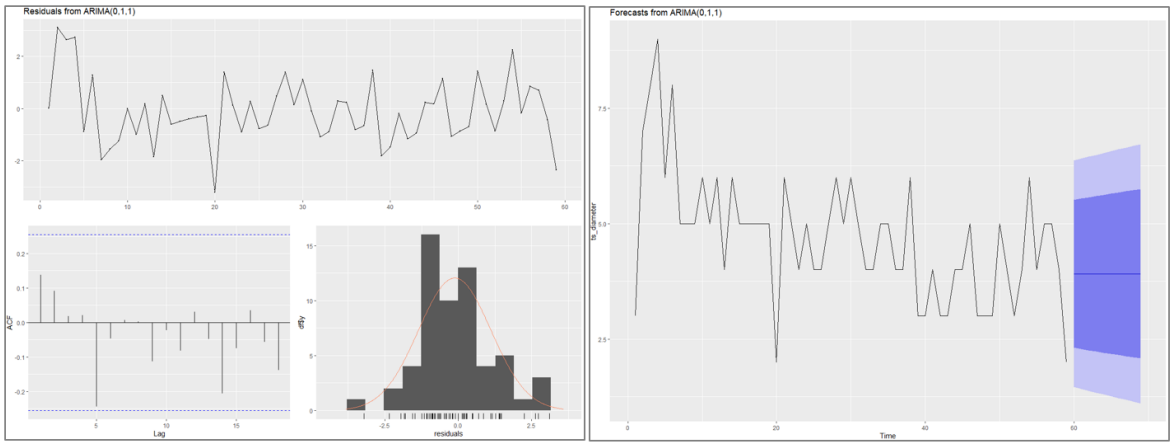


FIGURE 3.42 – ARIMA for diameter.

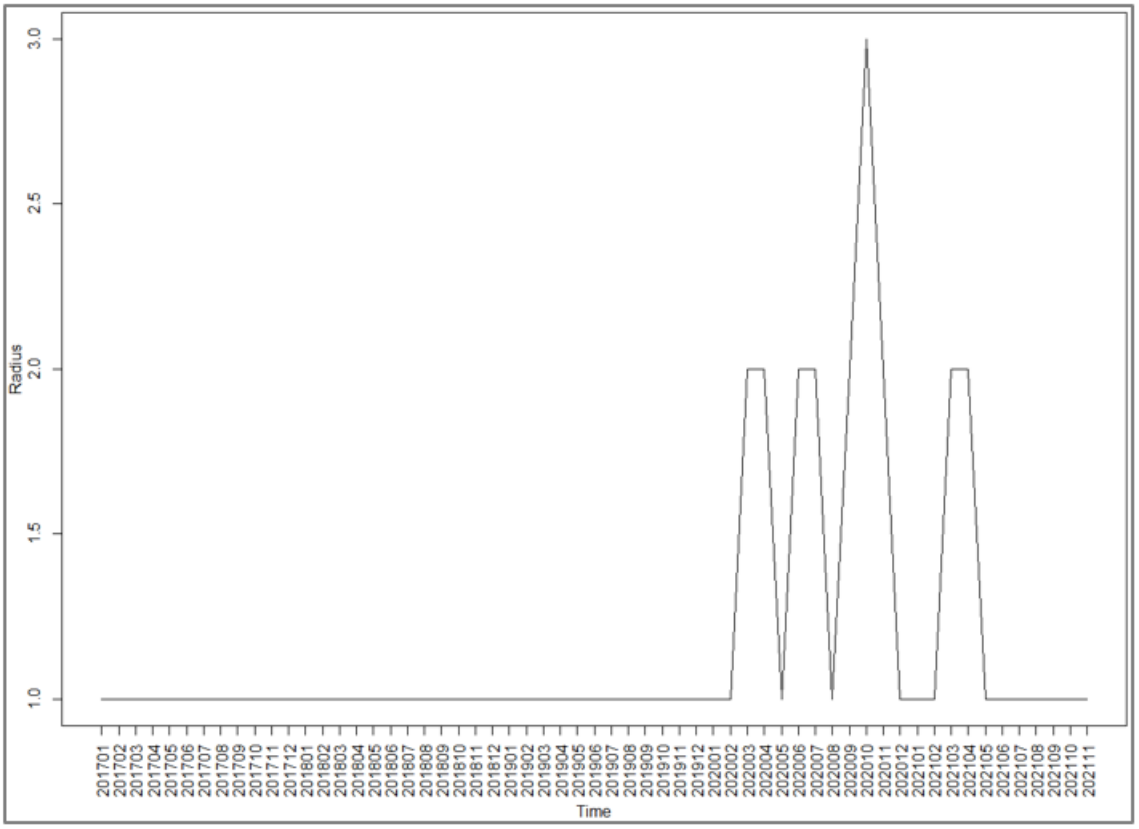


FIGURE 3.43 – Time series analysis for radius.

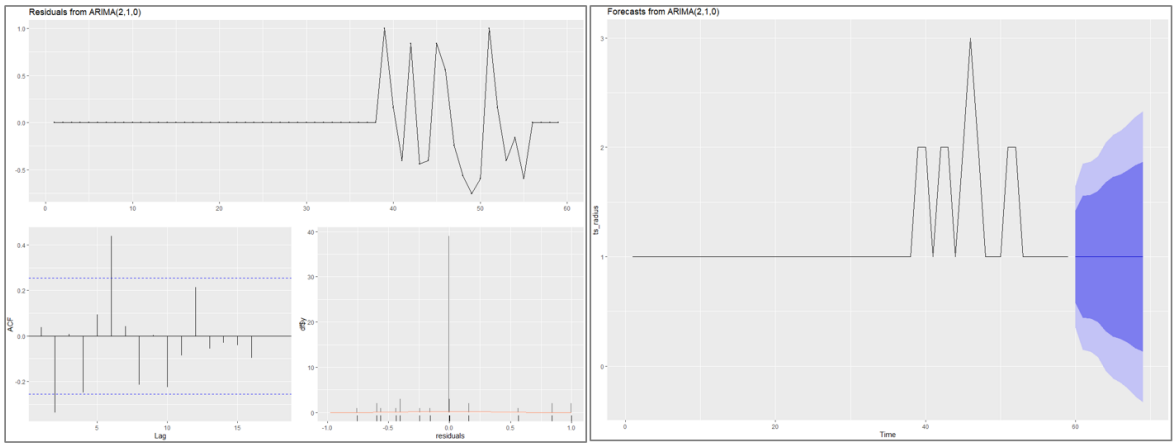


FIGURE 3.44 – ARIMA for radius.

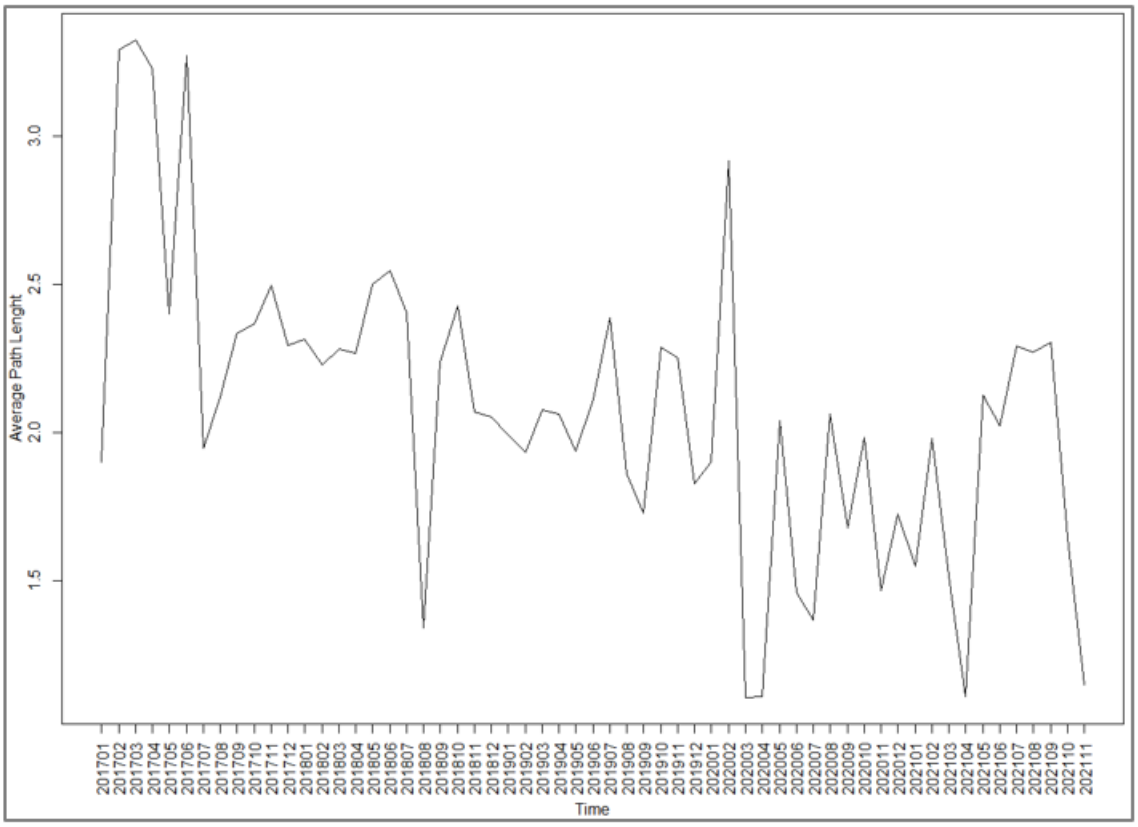


FIGURE 3.45 – Time series analysis for average path length.

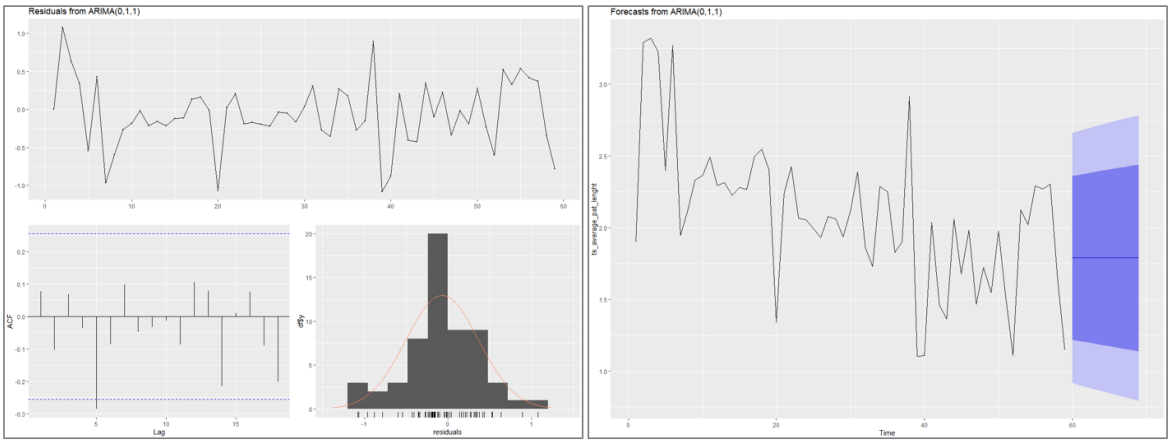


FIGURE 3.46 – ARIMA for average path length.

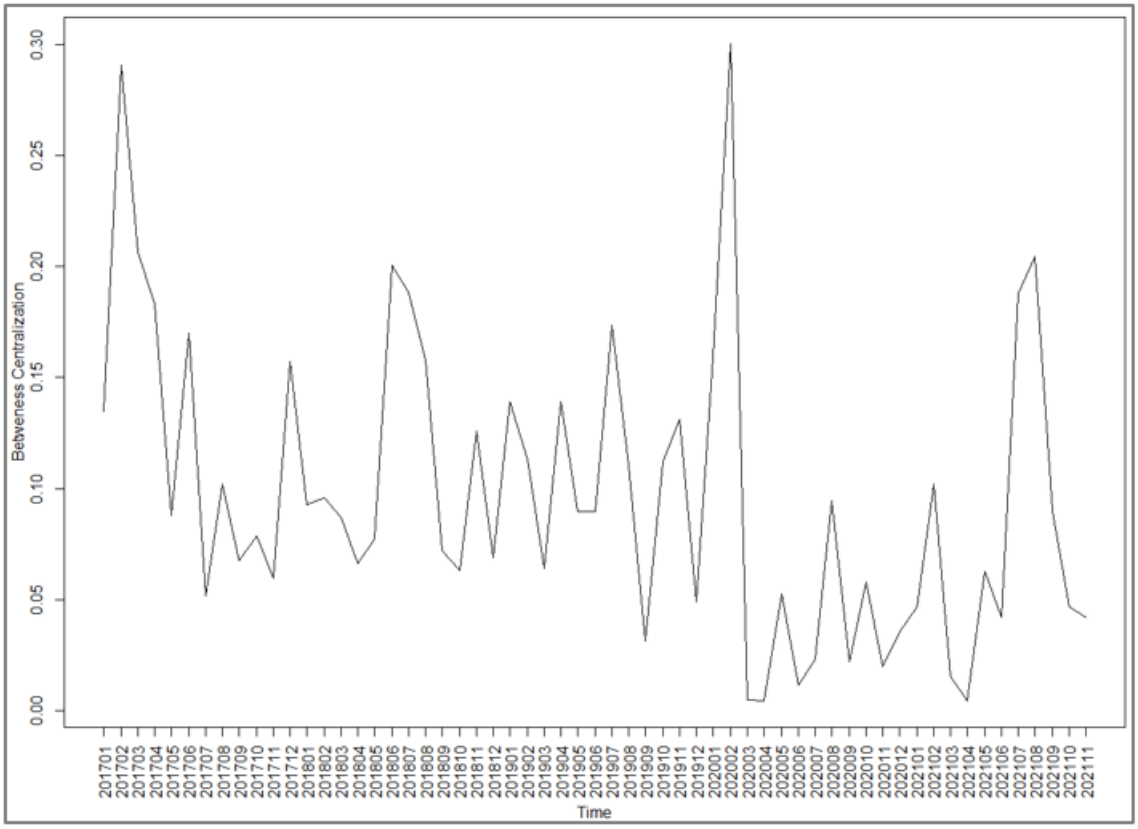


FIGURE 3.47 – Time series analysis for betweenness centralization.

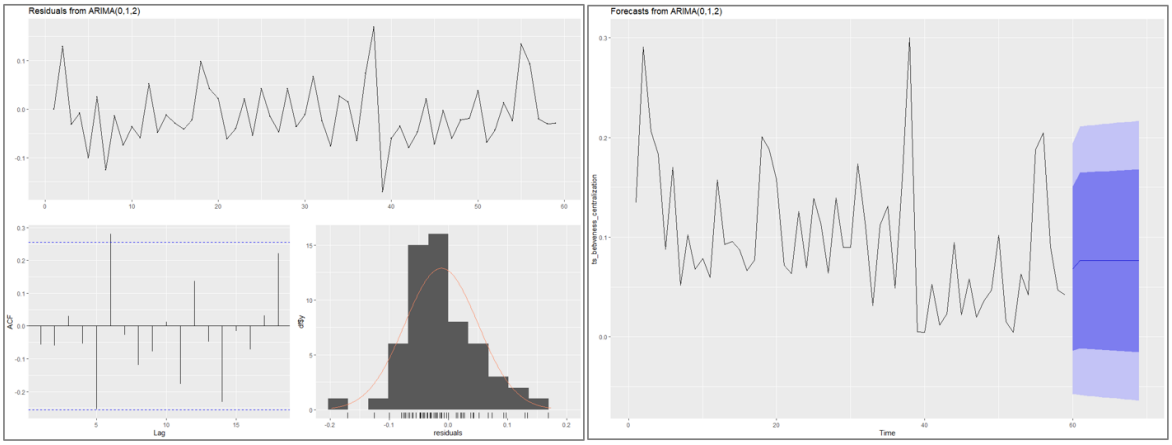


FIGURE 3.48 – ARIMA for betweenness centralization.

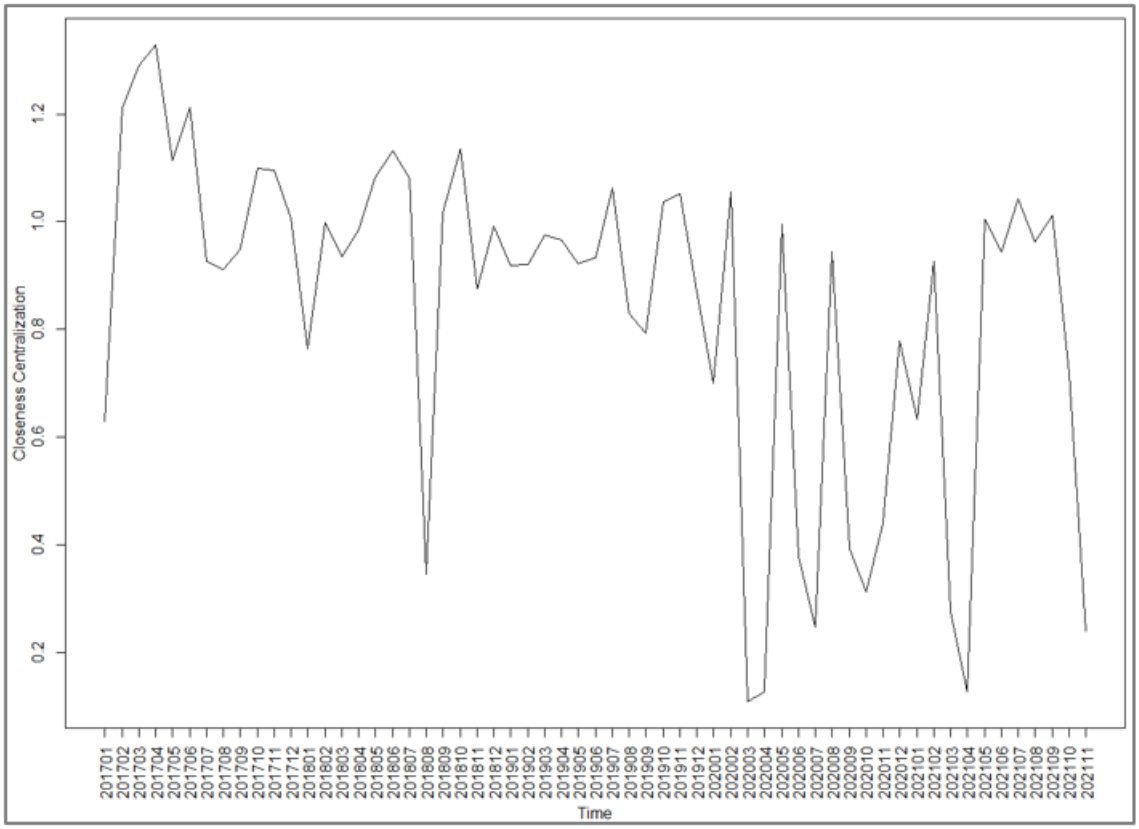


FIGURE 3.49 – Time series analysis for closeness centralization.

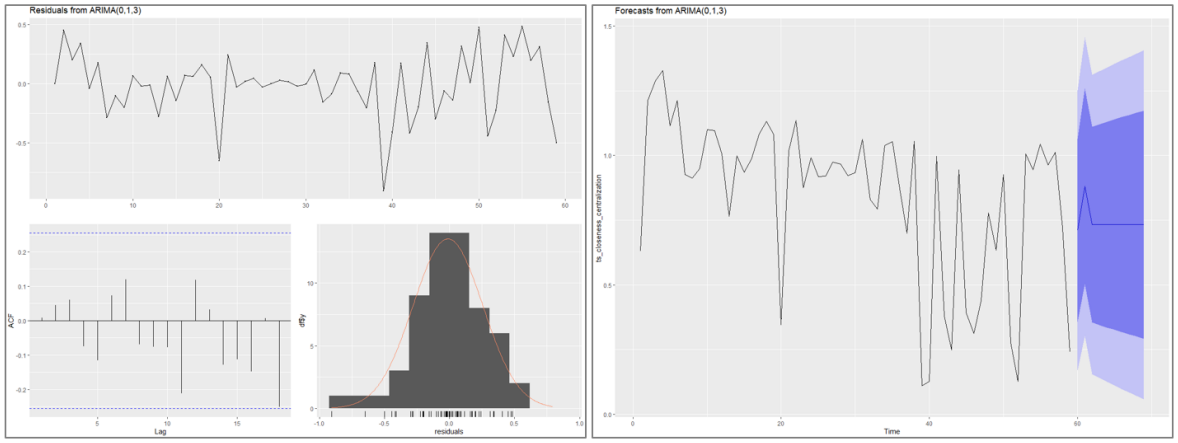


FIGURE 3.50 – ARIMA for closeness centralization.

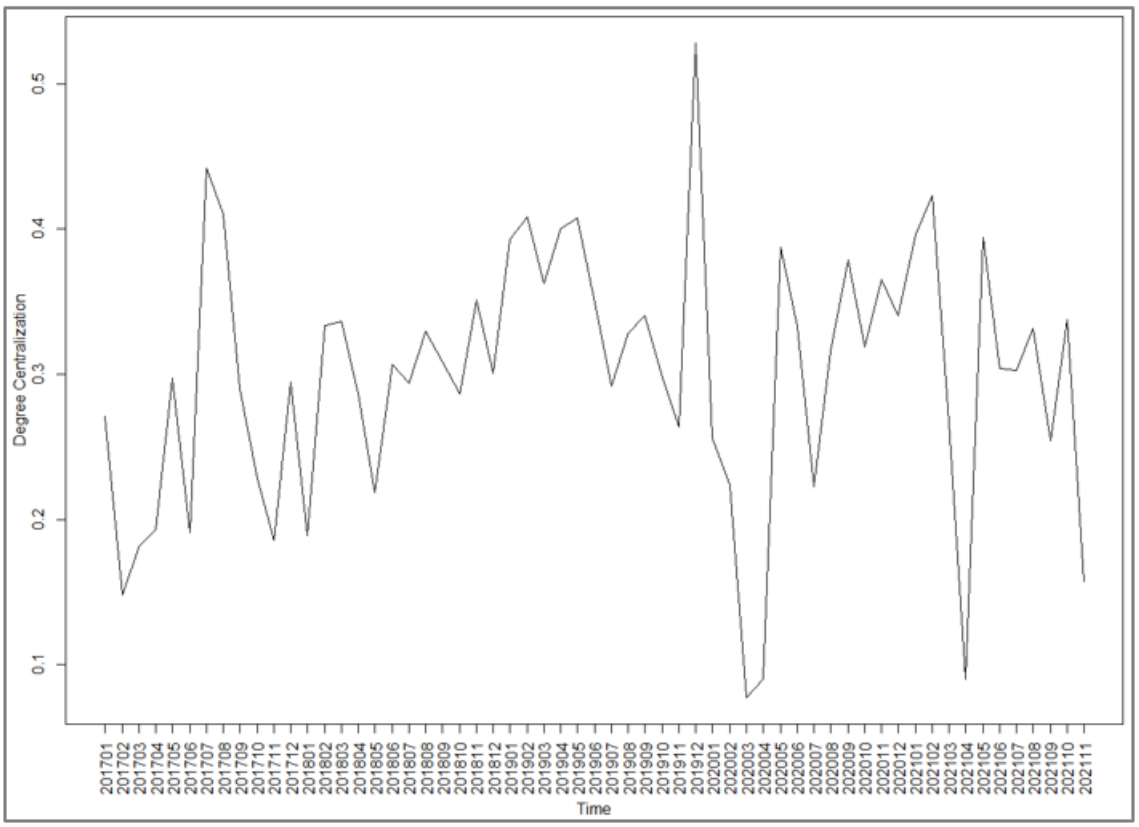


FIGURE 3.51 – Time series analysis for degree centralization.

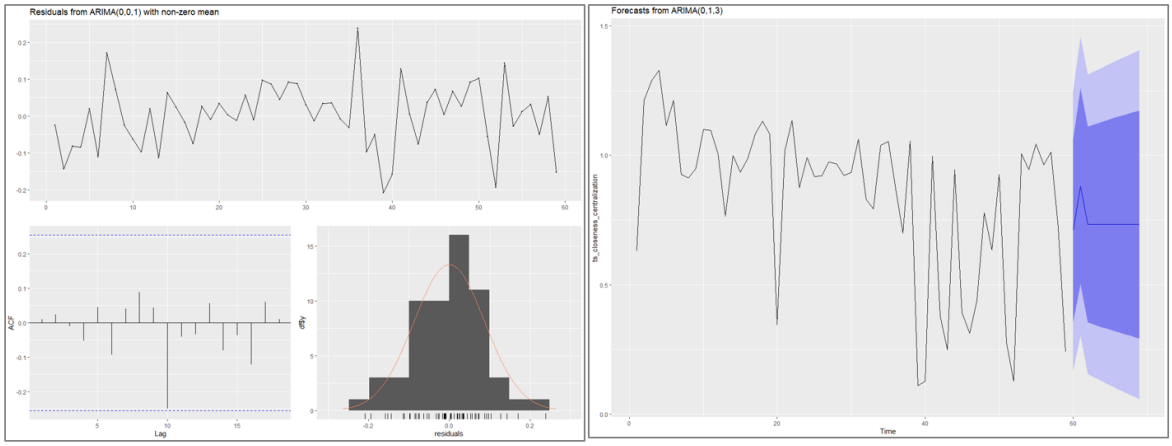


FIGURE 3.52 – ARIMA for degree centralization.

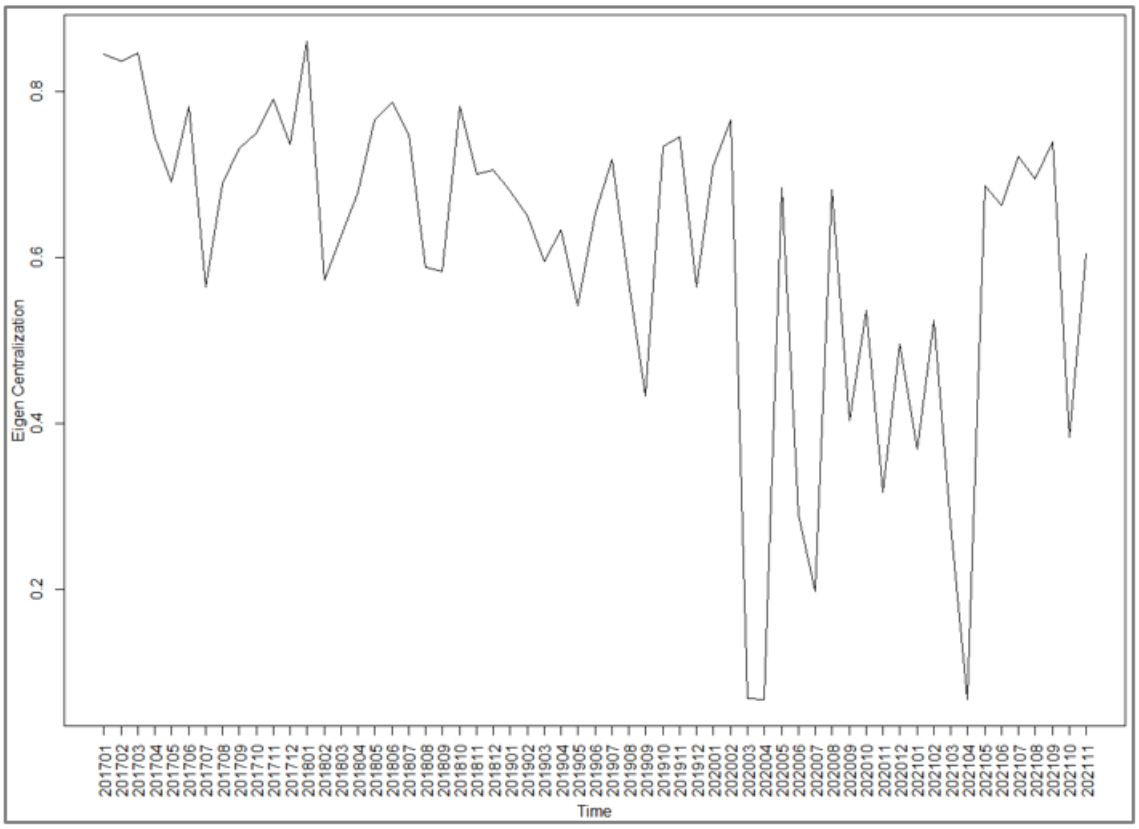


FIGURE 3.53 – Time series analysis for eigenvector centralization.

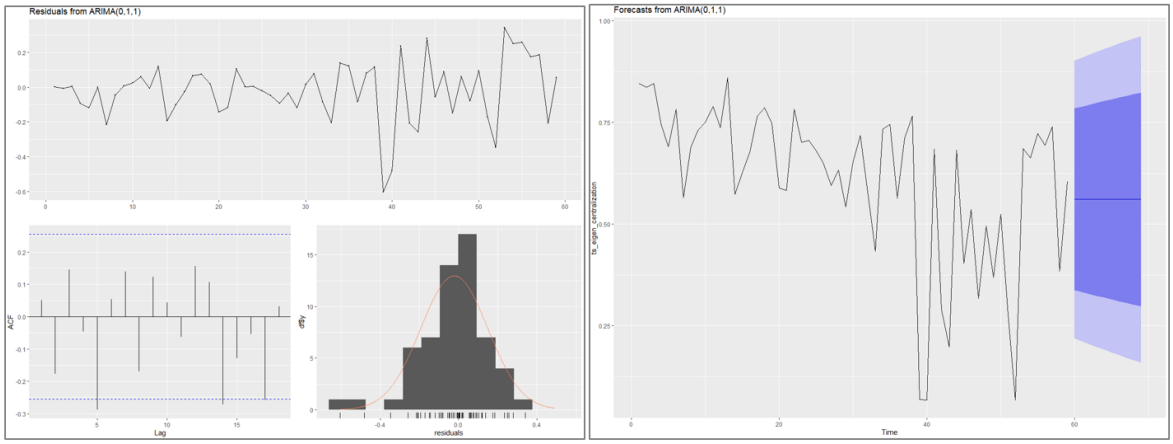


FIGURE 3.54 – ARIMA for eigenvector centralization.

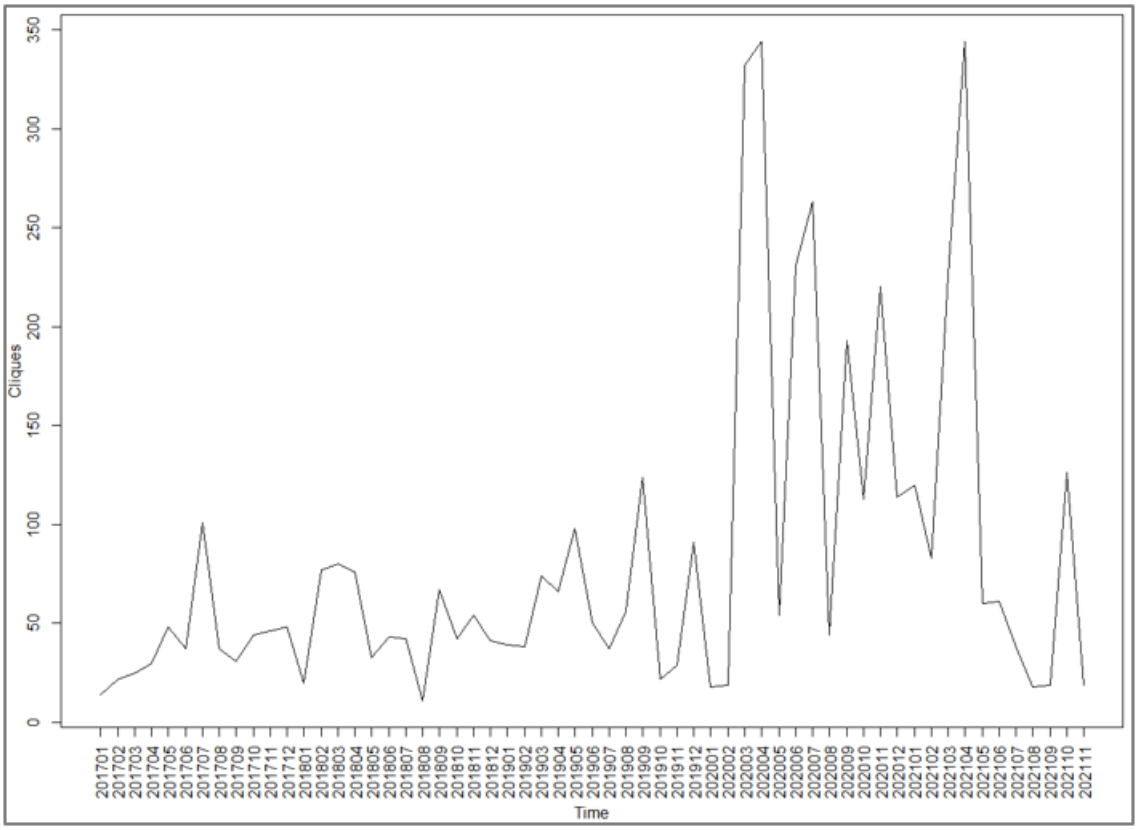


FIGURE 3.55 – Time series analysis for cliques.

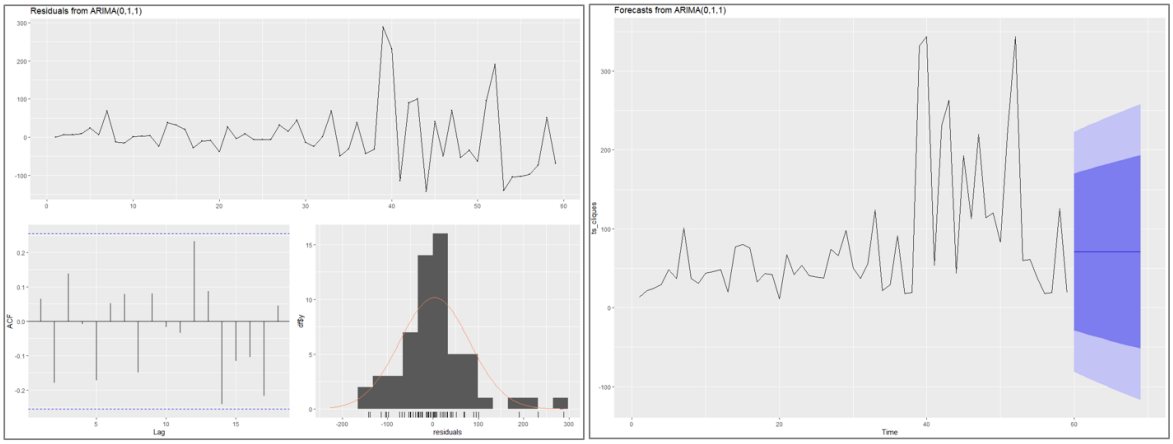


FIGURE 3.56 – ARIMA for cliques.

4 LINK PREDICTION RESULTS

4.1 Data Pre-processing

In this part of the study, all information about data sets and data pre-processing steps is explained from beginning to end. The first step involves providing information about the raw data set. Then, the process of creating the newly generated education dictionary is described. After these two steps, there is a section that provides information about the education data set, which is generated using the raw data and the education dictionary. The last step involves the generation of the education network.

4.1.1 About Raw Data Sets

In this study, the education data sets of all employees who worked continuously at Softtech between 2017 and 2021 are used. There are primarily two raw data sets. One of them is the "Online Education" data set, and the other is the "Offline and Virtual Class Education" data set. All information about the identification of employees is masked ; a user ID is used instead of an employee ID or employee name. There are 669 users in this study. The "Online Education" data set includes four columns that are named "User ID", "Education Name", "Sub Education Name" and "End Date of the Education". There are 24774 rows in that data set. The "Offline and Virtual Class Education" data set includes five columns that are named "User ID", "Education Name", "Sub Education Name", "Start Date of The Education" and "End Date of The Education". There are 16439 rows in that data set. "User ID" columns are in string format, and they sequentially increase one by one like UserId1, UserId2, UserId3, etc. "Education Name" and "Sub Education Name" columns are in string format, and they include different types of technical or non-technical education names. Some of the technical educations are like Java, C#, .NET Core, SQL, Python, Data Science, Machine Learning, Big Data, Deep Learning, HTML5/CSS3, Vue.js, NodeJS, Javascript, IoT, etc. Some of the non-technical educations are like Proactive Behavior, Planning and Organization, Time Management, Stress Management, Emotional Intelligence, etc. These educations may be given as examples. All date columns include day, month, and year information.

4.1.2 Generating Education Dictionary

First of all, there is a need to prepare an education dictionary. A new data set that is named "Education Dictionary" is generated. These data sets include two columns, "Education ID" and "Education Name". All 720 different educations are added to this dictionary, and therefore this data set includes 720 rows. The "Education ID" column is in string format, and it sequentially increases one by one, like Education1, Education2, Education3, etc. The "Education Name" column is in string format and it includes different types of technical or non-technical education names previously described. "Education Dictionary" would be used to convert education columns later.

4.1.3 Generating Merged and Transformed Education Data Set

"Online Education" and "Offline and Virtual Class Education" data sets are converted to the same format, and then they are merged to get only one "Education" data set. "Sub Education Name" columns are not used in the study, and so they are removed from the data sets. The "Start Date of the Education" column in "Offline and Virtual Class Education" is removed, and only the "End Date of the Education" columns in two data sets are used in this study by converting it to a general "Date" column. Only year information is extracted from "Date" columns; day and month information in "Date" columns are not taken into account in this study. "Education Name" columns are converted to "Education ID" by using the education dictionary. At the end, one merged "Education" data set that has "UserId", "EducationId", and "Year" columns is obtained from the "Online Education" and "Offline and Virtual Class Education" data sets.

4.1.4 Generating Networks

The "Education" data set is split into two parts : training and test data sets, according to the year information. If the year is smaller than 2021, the data is training data. If the year is 2021, the data is test data. After that operation, the year column is removed from training and test data sets. They include only "UserId" and "EducationId" columns; thus, training and test bipartite networks are got. The repetitive rows are deleted from training and test bipartite networks. The training bipartite network includes

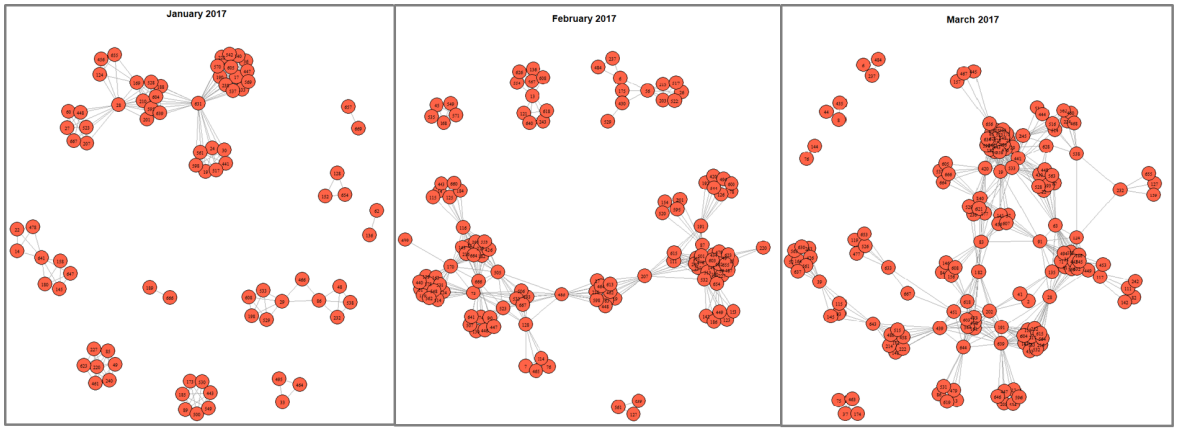


FIGURE 4.1 – Sample visualization of how employees are connected via education.

11459 different UserId-EducationId links, and the test bipartite network includes 2241 different UserId-EducationId links.

The users in bipartite networks are associated with each other using education information ; thus, bipartite networks are converted to training and test networks. The training network includes 119022 different UserId-UserId links, and the test network includes 64046 different UserId-UserId links. At the end, "User" string expressions are removed from the data sets, and thus numeric UserId-UserId links appear in the networks. All these training and test networks that include only numeric nodes are saved in edge files, and then the data pre-processing steps are ended. The users are nodes in the networks, and the educations are links in the networks.

Sample visualization of how employees are connected via education is shared in Fig. 4.1 to give an idea about network. The network is considered a dynamic network and includes only three months as a sample.

4.2 Link Prediction

In this part of the study, all link prediction features are calculated using the similarity and distance metrics. This process is applied separately to each network for all existing and missing connections.

The link prediction step begins with the preparation of training and test graphs. Firstly, the isolated nodes in the existing training and test networks are determined and saved. The training and test graphs are generated from the existing training and test networks

by separating the isolated nodes. The scores between two nodes are calculated for the new training and test graphs by using the link prediction methods that were previously mentioned. A file is generated for each link prediction method, so 15 files are generated for a graph because of the number of link prediction methods. Additionally, a label file is generated according to whether a link is visible or not for a graph. If the link already exists in the network, the value of the label is 1. Otherwise, the value of the label is 0. In total, these 16 files are generated for both training and test graphs separately. These files include node A, node B, and score columns. At the end, each training and test file is brought together among itself, and two main training and test data sets are generated for use in the machine learning step. These files include 21 columns and 222778 rows. The target column is the label column, and the other columns are the scores of link prediction methods and information about nodes.

A sample data set of how the training and test data sets include data is shared in Fig. 4.2 to give an idea of the output file. The data was selected randomly to show different cases.

	nodeA_original	nodeB_original	nodeA_Rid	nodeB_Rid	AA	CAA	CAR	CN	CPA	CRA	DICE	ISOMAP	JC	L3	LEIG	MCE	PA	RA	SPM	LABEL
1	34	477	34	469	93.69856	26452.45	306362	554	93862883214	547.7345	0.9848889	0.07220032	0.9702277	0.002535239	3.76E-05	3.89E-16	316350	1.722997	0.942457	1
2	477	514	469	500	93.52985	26367.95	305256	553	93189162345	545.987	0.9770318	0.01768951	0.955095	0.002540312	0.000152494	1.67E-16	320340	1.720324	0.9320995	1
3	34	514	34	500	93.52919	26367.69	305256	553	93184583370	545.9608	0.9901522	0.05675097	0.9804965	0.00253808	0.000189736	2.22E-16	311910	1.720277	0.9850289	1
4	36	477	36	469	93.52847	26370.53	305256	553	93186720161	545.9986	0.9839858	0.07076851	0.9684764	0.002533839	4.89E-05	4.72E-16	315780	1.720201	0.9840232	1
5	34	36	34	36	93.5278	26370.28	305256	553	93182141306	545.9724	0.9972949	0.003150572	0.9946043	0.002531259	8.59E-05	2.24E-16	307470	1.720153	1.060855	1
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
161469	302	616	300	577	4.190991	69.71937	600	25	602202	2.118226	0.1103753	0.9402651	0.05841121	0.000505226	0.000600148	0.7314374	11102	0.08825943	0.148826	1
161470	254	382	252	378	4.179739	72.42936	650	26	576693	1.293463	0.1870504	0.6260185	0.1031746	0.000467609	8.55E-05	0.7374422	13845	0.05173851	-0.001337026	0
161471	382	660	378	616	4.179739	72.42936	650	26	576693	1.293463	0.1870504	0.6260185	0.1031746	0.000467609	0.000203085	0.7374422	13845	0.05173851	0.003088033	0
161472	382	661	378	617	4.179739	72.42936	650	26	576693	1.293463	0.1870504	0.6260185	0.1031746	0.000467609	0.000146268	0.7374422	13845	0.05173851	0.002062323	0
161473	155	315	154	313	4.161871	60.74669	552	24	553429	2.079036	0.09795918	1.241876	0.05150215	0.000658598	0.0008614	0.7314309	15925	0.1061204	0.1416143	1
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
184916	251	372	249	369	0.1602493	0	0	1	12544	0	0.0041841	1.857355	0.002096436	5.53E-05	0.1757631	0.7412299	13021	0.001949318	0.001288549	0
184917	254	372	252	369	0.1602493	0	0	1	1792	0	0.0212766	1.855739	0.01075269	5.03E-05	0.1756743	0.0137746	1885	0.001949318	0.04368864	0
184918	323	372	321	369	0.1602493	0	0	1	11816	0	0.004424779	1.827728	0.002217295	5.43E-05	0.1758202	0.7412299	12267	0.001949318	0.01082583	0
184919	4	373	4	370	0.160249326	0	0	1	6412	0	0.004219409	1.537571989	0.002114165	7.36E-05	0.114023692	0.73518245	6885	0.001949318	0.010519931	0
184920	5	373	5	370	0.160249326	0	0	1	6258	0	0.004319654	1.486341103	0.002164502	7.34E-05	0.114052772	0.741136053	6720	0.001949318	-0.004045491	0
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
222774	611	650	663	668	0	0	0	0	0	0	0	999999	0	0	999999	999999	0	0	0	0
222775	624	650	664	668	0	0	0	0	0	0	0	999999	0	0	999999	999999	0	0	0	0
222776	647	650	665	668	0	0	0	0	0	0	0	999999	0	0	999999	999999	0	0	0	0
222777	648	650	666	668	0	0	0	0	0	0	0	999999	0	0	999999	999999	0	0	0	0
222778	649	650	667	668	0	0	0	0	0	0	0	999999	0	0	999999	999999	0	0	0	0

FIGURE 4.2 – Sample data set of how the training and test data sets include data.

4.3 Machine Learning

The output files generated in the previous step are used as input files in this step, and the process begins with data pre-processing in those files to run the machine learning algorithms. The label columns in training and test data sets are converted from integer to factor format, so they would be used as target columns. All null values are replaced with zero values in the training data set and the test data set. The machine learning

models only use the scores of link prediction methods and the values of the label columns. If the data types are not the same in the training and test data sets, they are converted to be the same. After the standardization process on the data, the machine learning algorithms are run. The results are obtained by running XGBoost, gradient boosting, random forest, logistic regression, support vector machine, and multilayer perceptron machine learning algorithms.

Firstly, only training and test data sets are used for running machine learning algorithms. Accuracy, precision, recall, and F1 scores are used to evaluate the performance of the trained models. In Table 4.1, the values of accuracy, precision, recall, and F1 scores are shown for each model. The best accuracy score is obtained by the support vector machine model. The gradient boosting and the random forest models have high accuracy scores, too. The XGBoost model has the lowest accuracy score. The gradient boosting model has the highest precision score, and the logistic regression model has the lowest precision score. The logistic regression model has the highest recall score, and the XGBoost model has the lowest recall score. The support vector machine has the highest F1 score, and the logistic regression model has the lowest F1 score. In general, the support vector machine model could be preferred.

TABLE 4.1 – First Link Prediction Results Via Machine Learning

Model	***Accuracy***	***Precision***	***Recall***	***F1 Score***
XGBoost	0.4424	0.9947	0.3396	0.5063
Gradient Boosting	0.9874	1	0.9580	0.9785
Random Forest	0.9756	0.9942	0.9265	0.9591
Support Vector Machine	0.9927	0.9945	0.9804	0.9874
Logistic Regression	0.8177	0.6120	1	0.7593
Multilayer Perceptron	0.8766	0.9960	0.7008	0.8227

In Fig. 4.3, all the confusion matrices that belong to each machine learning model are shown. The logistic regression model is successful at predicting true positive (TP) values. The support vector machine model is successful at predicting true negative (TN) values. The gradient boosting model is successful at predicting the true positive (TP) and true negative (TN) values, too.

Then this experiment is repeated with a new validation data set. The training set is

XGBOOST

	Reference	
Prediction	0	1
0	34839	123893
1	339	63707

GRADIENT BOOSTING

	Reference	
Prediction	0	1
0	155921	2811
1	2	64044

RANDOM FOREST

	Reference	
Prediction	0	1
0	153678	5054
1	371	63675

SUPPORT VECTOR MACHINE

	Reference	
Prediction	0	1
0	157459	1273
1	351	63695

LOGISTIC REGRESSION

	Reference	
Prediction	0	1
0	118125	40607
1	0	64046

MULTI LAYER PERCEPTRON

	Reference	
Prediction	0	1
0	131495	27237
1	257	63789

FIGURE 4.3 – Confusion matrices of the first experiment.

divided into two parts at a rate of 30% to 70% randomly and homogeneously. 30% of the training data set is used as validation data, and 70% of the training data set is used as new training data. The machine learning algorithms are run again with a new training data set. The experiment is repeated on both validation and test data. This second experiment is necessary to understand whether the models are overfitting or not. The results are shown in Table 4.2. According to these comparative experiments, the algorithms are not overfitting. Again, the best scores are generally obtained by support vector machine. It can be said that making predictions using an education network is a logical method.

TABLE 4.2 – Second Link Prediction Results Via Machine Learning

Model		***Accuracy***	***Precision***	***Recall***	***F1 Score***
XGBoost	**Validation**	0.9993	0.9992	0.9996	0.9994
	Test	0.3684	0.9955	0.3123	0.4754
Gradient Boosting	**Validation**	0.9962	0.9962	0.9967	0.9965
	Test	0.9879	1	0.9596	0.9794
Random Forest	**Validation**	0.9998	0.9999	0.9998	0.9998
	Test	0.9717	0.9949	0.9143	0.9529
Support Vector Machine	**Validation**	0.9981	0.9976	0.9987	0.9982
	Test	0.9932	0.9945	0.9822	0.9883
Logistic Regression	**Validation**	0.9988	0.9991	0.9986	0.9989
	Test	0.8177	0.6120	1	0.7593
Multilayer Perceptron	**Validation**	0.9985	0.9982	0.9989	0.9986
	Test	0.8696	0.9965	0.6889	0.8146

The confusion matrices that belong to the new experiment are shown in Fig. 4.4. It is possible to make similar comments with previous results. The logistic regression model is successful at predicting true positive (TP) values. But it is not successful at predicting true negative (TN) values. The support vector machine model is successful at predicting true negative (TN) values.

XGBOOST

	Reference	
Prediction	0	1
0	18323	140408
1	290	63756

GRADIENT BOOSTING

	Reference	
Prediction	0	1
0	156036	2696
1	2	64044

RANDOM FOREST

	Reference	
Prediction	0	1
0	152759	5973
1	328	63718

SUPPORT VECTOR MACHINE

	Reference	
Prediction	0	1
0	157577	1155
1	350	63696

LOGISTIC REGRESSION

	Reference	
Prediction	0	1
0	118125	40607
1	0	64046

MULTI LAYER PERCEPTRON

	Reference	
Prediction	0	1
0	129905	28827
1	222	63824

FIGURE 4.4 – Confusion matrices of the second experiment.

As a result of these experiments, we can accurately predict whether there is a link, according to the education that they received, between the two employees. For an employee, an education plan can be generated from all education received by the employees that employee is related. All education that is received by non-related employees can be excluded from the priority of this plan. The final plan can be created after all education that has been received by the employee until that date is also removed from the plan. In this study, we do not keep education information on the link. In the next stage of the study, education information could be recorded on the link, and a direct education prediction would be made. According to the number of joint educations received, it is possible to take into account the strength of the bond between the employees, too.

In addition to these experiments, the reason why the XGBoost algorithm yields lower results compared to other algorithms is treated as a separate experiment. In the conference paper associated with this thesis, all algorithms provided approximate results. In the experiments conducted in the thesis study, the reasons for the low performance

of the XGBoost algorithm are examined. The main difference between the thesis study and the conference paper is that the conference paper utilizes a greater number of link prediction methods. In addition to the link prediction methods used in the thesis study, 14 additional different link prediction techniques have been employed. These different methods are as follows :

1. Random Walk with Restart (RWR)
2. Matrix Forest Index (MF)
3. Local Paths Index (LP)
4. Leicht-Holme-Newman Index (LHN LOCAL)
5. LeichtHolme-Newman Global Index (LHN GLOBAL)
6. Pseudoinverse of the Laplacian (L)
7. Katz Index (KATZ)
8. Hub Promoted Index (HPI)
9. Hub Depressed Index (HDI)
10. Geodesic distance vertex similarity (DIST)
11. Cosine Similarity based on the pseudoinverse of the Laplacian Matrix (COS L)
12. Cosine vertex similarity/ Salton index (COS)
13. Average Commute Time (ACT)
14. Average Commute Time, normalized (ACT N)

So, in the thesis study, there are 15 features excluding the label column, while in the conference paper, there are 29 features excluding the label column. The new experiment is repeated using only the XGBoost algorithm, utilizing these different data sets. Feature selection is performed in the experiments.

In the first experiment, the label column is predicted using 15 features. Since the results have been the same as the previous ones, a new table is not generated. The value of the accuracy is 0.4424. The value of precision is 0.9947. The value of recall is 0.3396. The value of the F1 score is 0.5063. The importance matrix has been obtained for the first experiment. The results are shown in Table 4.3.

TABLE 4.3 – The importance matrix for first experiment

	Feature	***Gain***	***Cover***	***Frequency***
1	SPM	9.916391e-01	3.236562e-01	0.381231672
2	AA	2.274838e-03	9.764720e-02	0.149560117
3	L3	1.868396e-03	2.835132e-01	0.184750733
4	CAA	1.590952e-03	4.845796e-02	0.038123167
5	DICE	1.411126e-03	4.054945e-02	0.061583578
6	CRA	5.761922e-04	5.795577e-02	0.058651026
7	RA	3.201407e-04	8.514372e-02	0.067448680
8	CPA	2.402476e-04	3.542113e-02	0.026392962
9	PA	7.693533e-05	2.012874e-02	0.023460411
10	LEIG	1.819262e-06	7.525380e-03	0.005865103
11	ISOMAP	2.774247e-07	1.321359e-06	0.002932551

The matrix results are visualized as shown in in Fig. 4.5. SPM is more dominant compared to the others. However, in general, there are no very high values.

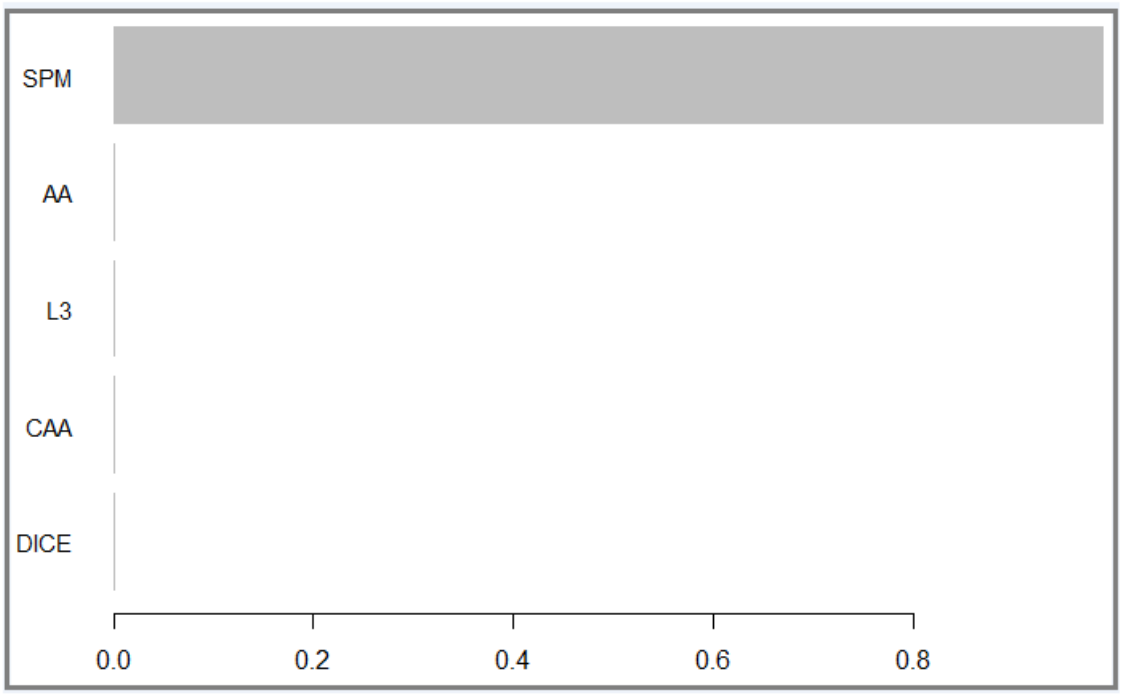


FIGURE 4.5 – Visual of feature importance for first experiment.

In the second experiment, the label column is predicted using 29 features. Since the results have been the same as the previous ones, a new table is not generated. The value of the accuracy is 0.9989. The value of precision is 1. The value of recall is 0.9963. The value of the F1 score is 0.9981. The importance matrix has been obtained for the second

experiment. The results are shown in Table 4.4.

TABLE 4.4 – The importance matrix for second experiment

	Feature	***Gain***	***Cover***	***Frequency***
1	KATZ	1	1	1

The matrix results are visualized as shown in Fig. 4.6. There is only KATZ, and all its values are 1.

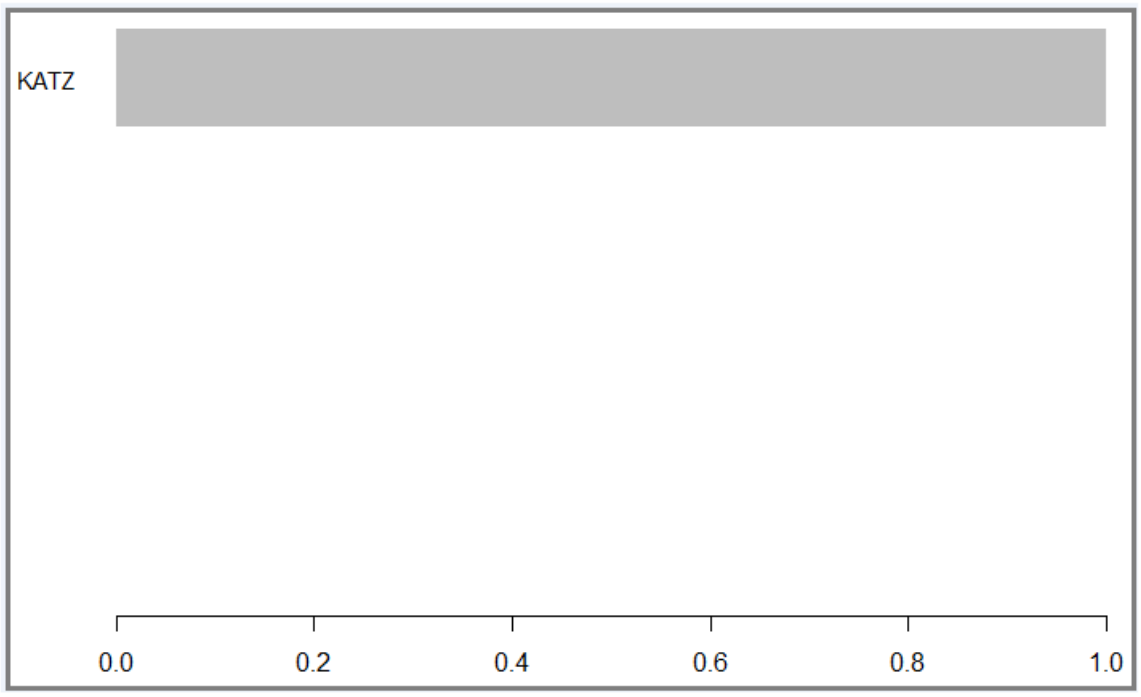


FIGURE 4.6 – Visual of feature importance for second experiment.

The XGBoost algorithm is a powerful algorithm. From this experiment, we observe that using a powerful algorithm on small data sets can lead to overfitting, and such risks are possible.

5 CONCLUSION

In this study, the education data sets of Softtech employees have been used. Softtech is a software engineering company in Turkey, as mentioned before, and these data sets were created specifically for this study. This study begins with the education records that belong to 2017, because the education data sets of the employees have been recorded regularly since 2017. There are five years of data available, from the beginning of 2017 until the end of 2021. These data sets have been converted to a network after the data pre-processing steps. The employees are connected with each other via education networks. Masked employee information, the education they received, and the dates of the education are sufficient for the study.

Firstly, the network has been considered and analyzed as a dynamic network. The dynamically generated networks have been analyzed with macro, micro, meso, and time series analysis techniques on a monthly basis and visualized. The impacts of the COVID-19 pandemic on the data have been observed. Then the network has been considered a static network, and the link prediction processes have been applied to it. A link prediction framework for education recommendations to employees has been proposed. At the end of the link prediction step, new data sets have been created for use as input files in the machine learning step. XGBoost, gradient boosting, random forest, logistic regression, support vector machine, and multilayer perceptron machine learning algorithms have been run. Accuracy, precision, recall, and F1 scores have been used for evaluating the performance of the machine learning models.

After creating networks using the data, it has been observed that the data is suitable for network formation and the conducted study is reasonable. When the network is considered and examined as a dynamic network, some results have been obtained from the changes in the network. At the company, annual, quarterly-based business planning is conducted, and business objectives are set accordingly. This situation has influenced employees willingness to receive education and, consequently, their educational connections with each other. It can be said that during periods of high workload, the training network becomes sparse, while during periods of low workload, the network density is higher. Additionally, it is observed that during holiday or collective leave periods,

there is a decrease in the number of individuals receiving training and the number of training connections. The results of macro analysis studies have been supported by the generated network visualizations. The network visualizations have already been created based on these results.

When detailed network visuals are examined, it has been observed that some employees differ in terms of receiving training. There are noteworthy results in terms of the number of nodes, the number of connections, and the structure of connections that are worth examining. Even without this study, it is possible to determine which employees receive more training, which education they have taken, and so on. However, when the connections established by these employees with other employees through education and the community structures in the resulting networks are examined, it directs us towards different studies that can be conducted. In this study, information about the employees' positions, previous work experiences, workload, etc. is not included. Considering this information and conducting a new study can lead to further research on determining career paths based on the education received by employees. For example, employees with high betweenness centrality serve as bridges between different training groups. When we remove these individuals from the network, the network group will disintegrate, and the network structure will change. These individuals, who serve as bridges between different training groups, may be considered candidates for team leader or product manager positions in teams consisting of people with diverse training and skills. The employee's diverse educational interests and the education courses he/she has taken indicate that the groundwork is ready for this. Similarly, it is possible to make similar interpretations for individuals with high eigenvector centrality. It is possible to facilitate the transformation of a newly hired employee or someone newly added to a team to adapt and serve the company or the team as desired through education. This study can be utilized to achieve these transformations more intelligently with the right training plan. By tracking the positions and transformations of similar individuals in the network structures, we can make more realistic and rational education recommendations and plans when positioning an employee within the network. Micro-level analyses will be useful in this context. As already observed in the degree distribution graphs, it has been found to generally exhibit a power-law distribution. This situation is evidence that it is a real-life example because networks in real life also exhibit power-law distributions.

As a result of the meso-level analyses conducted, community detection processes have been performed. Different algorithms have been tried, and it has been observed that all methods yielded almost the same results. Only minor differences are present. Subsequently, using the generated heatmap, it was determined which algorithms produced more similar results to each other. In the study conducted for January 2017, the Leading Eigenvector algorithm shows differentiation compared to the others. As seen in the visualization, while the other algorithms divide the large community into four parts, the Leading Eigenvector algorithm divides it into three parts. Infomap, Edge-Betweenness, and Walktrap yielded the same results. The community detection process was primarily initiated in this study to determine whether community analysis can be performed on dynamic networks. However, later in the time series analyses, it was observed that due to the changing and inconsistent structure caused by the COVID-19 pandemic's impact, this study was abandoned. In time series analysis studies, there has been a clear increase in the number of employees receiving education (nodes) and the number of educations received (links) since the start of the COVID-19 pandemic. All parameters related to network analysis have been examined in this context. However, the abnormal increases due to the impact of the COVID-19 pandemic have hindered making predictions using the time series analysis method.

As a result of all these studies, it was decided to proceed with the link prediction method in the research. Since the network will be used as a static network, data preprocessing and network generation processes have been carried out in this context. Fifteen different link prediction methods have been used to predict whether there are connections between employees based on the education they have received. This prediction process has been performed using six different machine learning algorithms. In general, all algorithms have provided the same result. Due to its consistency, the support vector machine algorithm can be preferred. The high accuracy values of the algorithms indicate that our data and network are suitable for this study. It has been demonstrated that performing link prediction using the employee education data set is feasible. It has been observed that the XGBoost algorithm provides significantly lower results compared to other algorithms. This situation has also been further examined. Based on this additional experiment, it has been observed that the XGBoost algorithm gives low accuracy values on small data sets due to overfitting. Powerful models are prone to overfitting ; this is formalized in the bias-variance tradeoff.

Ultimately, a model was constructed for the education recommendation system, and it was observed that this model could be utilized. Employees working at the same company are integral parts of the same system, and even if they don't know each other, they share similar educational interests and preferences. Therefore, it is possible to present these preferences to employees. In this way, the education recommendations made provide employees with more engaging suggestions compared to traditional methods. In this study, accurate link predictions are obtained by combining different features in the machine learning model. The addition of educational attributes to the study can be complementary and enhance its completeness. In the continuation of the study, comparing the obtained analytical results with real-life outcomes will help ground our analytical findings on a practical basis. The current results show the feasibility of this study. But as the amount of data increases, the realistic nature of the results will increase proportionally. It is also possible to get more realistic results by detailing the education information in the links. The next steps can be guided by employees' feedback based on education recommendations. Another alternative is to compare the results of recommended and preferred educations in real life.

In today's discussions about data-driven human resources, data-driven recruitment techniques, etc., it has become inevitable to plan employee education based on data, provide education recommendations to employees, and shape employees through education. Learning analytics, especially adult learning analytics, serves this purpose. In this thesis study, we have made a new contribution to the literature by using the employee education network for connection prediction, going beyond the conventional methods.

REFERENCES

Adamic, L. and Adar, E. (2003). Friends and neighbors on the web, *Social Networks* **25**(3) : p. 211–230.

Belkin, M. and Niyogi, P. (2002). Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering, *in* T. G. Dietterich, S. Becker and Z. Ghahramani (eds), *Advances in Neural Information Processing Systems 14*, MIT Press, pp. 585–591.

Cannistraci, C., Alanis-Lobato, G. and Ravasi, T. (2013a). Minimum curvilinearity to enhance topological prediction of protein interactions by network embedding., *Bioinform.* **29**(13).

Cannistraci, C. V., Alanis-Lobato, G. and Ravasi, T. (2013b). From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks, *Scientific Reports* **3**.

Chen, B. and Poquet, O. (2022). Networks in learning analytics : Where theory, methodology, and practice intersect, *Journal of Learning Analytics* **9**(1) : p. 1–12.

Dice, L. (1945). Measures of the amount of ecologic association between species, *Ecology* **26**(3) : p. 297–302.
URL: *http ://www.jstor.org/pss/1932409*

H., R. J. and A., G. (2018). *Forecasting : Principles and Practice*, second edn, OTexts.

Jaccard, P. (1912). The distribution of the flora in the alpine zone, *New Phytologist* **11**(2) : p. 37–50.

Kovács, I., Luck, K., Spirohn, K., Wang, Y., Pollis, C., Schlabach, S., Bian, W., Kim, D., Kishore, N., Hao, T., Calderwood, M., Vidal, M. and Barabási, A. (2019). Network-based prediction of protein interactions, *Nature Communications* **10**(1).

Kuchaiev, O., Rasajski, M., Higham, D. and Przulj, N. (2009). Geometric de-noising of protein-protein interaction networks, *PLoS Comput. Biol.* **5**(8).

Lü, L., Pan, L., Zhou, T., Zhang, Y. and Stanley, H. (2015). Toward link predictability of complex networks, *Proceedings of the National Academy of Sciences* **112**(8) : p.

2325–2330.
URL: *https ://www.pnas.org/doi/abs/10.1073/pnas.1424644112*

Newman, M. (2001). Clustering and preferential attachment in growing networks, *Physical Review E* **64**(2) : p. 025102.

softtech.com.tr (n.d.). Who are we ?
URL: *https ://softtech.com.tr/en/about-us/*

Tatel, C. E., Lyndgaard, S. F., Kanfer, R. and Melkers, J. E. (2022). Learning while working : : Course enrollment behaviour as a macro-level indicator of learning management among adult learners, *Journal of Learning Analytics* **9**(3) : p. 104–124.

Zhou, T., Lü, L. and Zhang, Y. (2009). Predicting missing links via local information, *The European Physical Journal B-Condensed Matter and Complex Systems* **71**(4) : p. 623–630.

BIOGRAPHICAL SKETCH

Name and Surname : Ceyda Kocaman

Birthday /Birthplace :

Education :

Master of Science in Smart Systems Engineering (Thesis), Galatasaray University, August 2023 (Expected)

Master of Science in Big Data Analytics and Management (non-Thesis), Bahcesehir University, January 2018

Bachelor of Science in Mathematical Engineering, Yildiz Technical University, September 2011

Naim Atakas Anatolian High School, June 2007