



# COS70008- TECHNOLOGY INNOVATION PROJECT: SOCIAL NETWORK ANALYSIS

Project Report

Name: Kritika.

# Introduction

## Project Background And Motivations

The project aimed to develop an interactive and user-friendly web application which will be able to perform social network analysis (SNA) with the integration of statistical models like exponential random graph models (ERGMs) or auto-logistic actor attribute models (ALAAMs). Users should be able to collect and upload network data, perform descriptive network statistics, visualize the network, perform statistical analysis using models like ERGMs or ALAAMs. A user can also generate reports to further study the data and benefit from it. The application aims to empower researchers, data analysts and social scientists with an efficient and interactive tool for social network dynamics.

## Design concept:

### Problem statement

The main objective of the project is to create an interactive and comprehensible tool for social network analysis (SNA) which will be able to upload/collect data, process and validate network data within the application itself. There should be integration of statistical models like ERGMs or ALAAMs along with network visualization capabilities to enable users to explore network structure. The tool should be able to provide a user-friendly interface and along with interactive report generation facility which will provide users with network data insights and practical applications of the analysis of the results.

## Design Specifications

- **Data handling:** Uploading of data in csv or excel formats, validation, and processing of data using appropriate libraries making it compatible to perform other operations.
- **Network statistics:** Calculation of descriptive network visualization metrics like degree centrality, closeness centrality, eigenvector centrality, etc. and network statistical metrics like number of edges, number of nodes, average degree, etc. Ability given to the user to choose the desired metrics.
- **Network Visualization:** Visualization of the network graphs for better understanding of network structure and interactivity of the network graphs make it easier to comprehend.
- **Statistical models:** Integration of statistical models like ERGMs or ALAAMs for deeper analysis into the network data. Calculation of goodness of fit (GOF) for the statistical models implemented.
- **Report generation:** Report generating functionalities implemented in HTML or excel formats to help users study and analyse the data and benefit from it.

## Design Implementation, Processes (Hardware and Software)

- Our design concept leverages ‘Streamlit’ to host the web application and for smooth integration of all the functionalities and to keep the track of the project progress a ‘GitHub’ repository was created. ‘Streamlit’ works with python language and convert the code scripts to shareable web applications and is time efficient in doing so. [1]

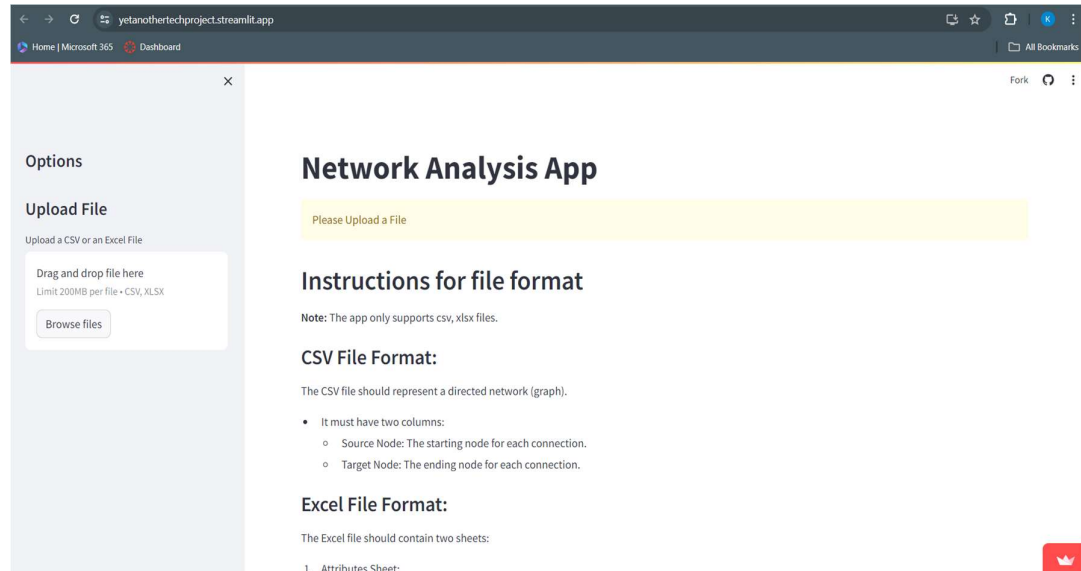


Figure 1: Home page of the web application

- Data can be uploaded in excel or csv formats. The files will be holding either network data having source or target columns or a dataframe having attributes, based on which our statistical models will act. This is with respect to the sample data provided.
- Our web application is based on ‘Python’, we use python libraries like ‘pandas’ for data ingestion and data processing. For providing network statistical metrics and for visualization metrics we used ‘NetworkX’ python package. This package used for creation, manipulation, and analysis of complex graph networks. It has various functions like giving data structures for different types of graphs, having various graph algorithms, giving network statistics measures and so on. [2]
- For creating interactive and user-friendly network graph visualizations we used ‘PyVis’ which is python-based package. It allows each graph to be interactive and allow selection of edges or nodes. It is built using a JavaScript library called ‘VisJS’.[3]



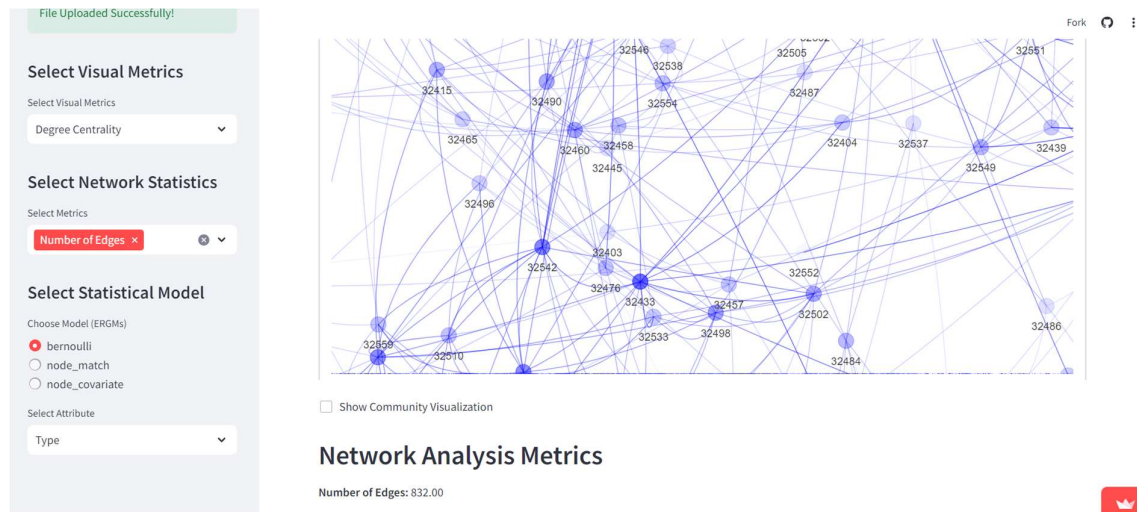


Figure 4: Choosing the network statistics metrics.

- Further for statistical analysis we implemented ERGM model, and we were able to do that based on the network edges (Bernoulli), node covariance and node match functions.

ERGMs check whether two nodes will have a tie between them or not, they are analogous to the standard logistic regression methods [4]. This was done using both R and Python languages.

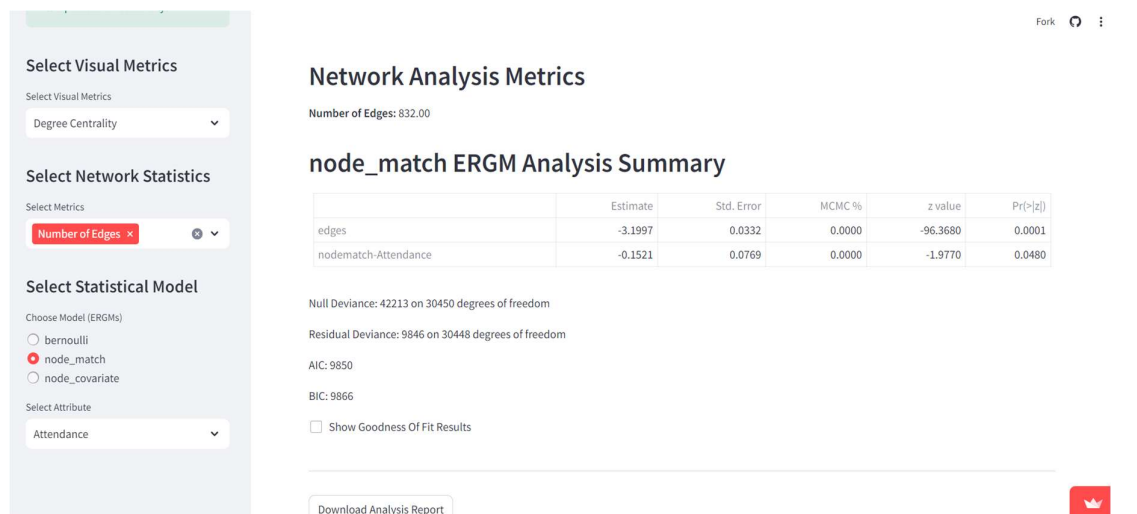


Figure 5: Statistical model functionality with an option to check goodness of fit results.

- Then for both network statistics and statistical analysis results a user will be able to generate and download reports, ideally in HTML and excel formats.

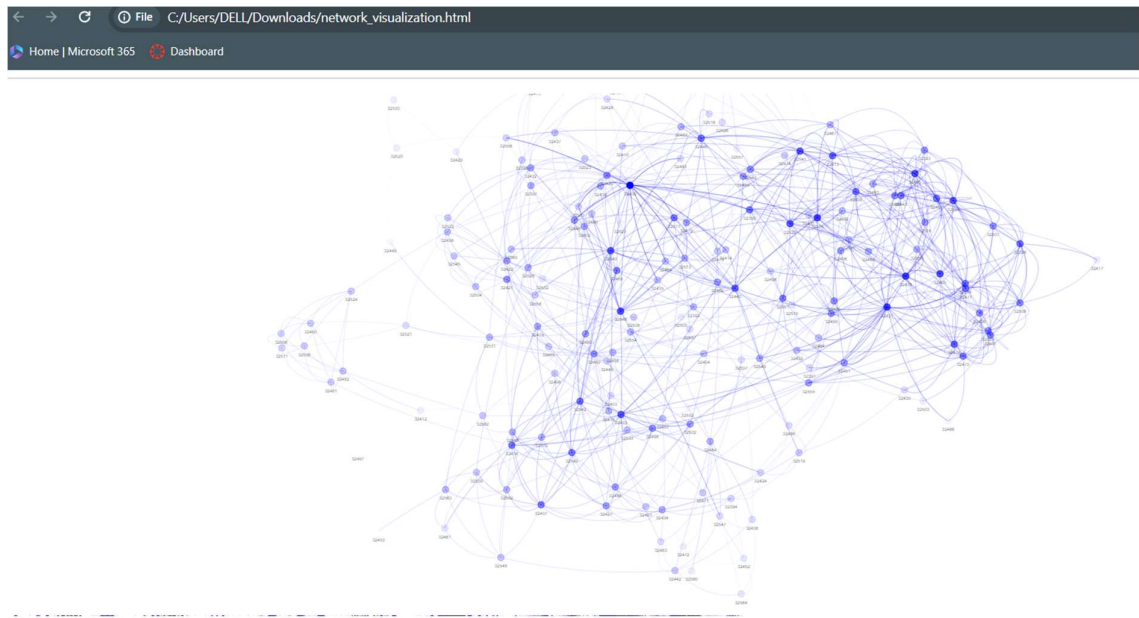


Figure 6: HTML report generated.

Network Analysis - Protected... • Saved to this PC													
Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View. <span>Enable Editing</span>													
Node													
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Node	Community	Free Centrality	Centrality	Centrality	Centrality	PageRank	Hub Score	Authority	Scores			
2	32561	7	0.005747	0.232932	0	0.000259	0.001551	3.15E-05	3.15E-05				
3	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
4	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
5	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
6	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
7	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
8	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
9	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
10	32531	0	0.051724	0.343195	0.025601	0.007249	0.006613	0.00088	0.00088				
11	32444	4	0.04023	0.324627	0.011918	0.016966	0.005611	0.002061	0.002061				
12	32444	4	0.04023	0.324627	0.011918	0.016966	0.005611	0.002061	0.002061				
13	32444	4	0.04023	0.324627	0.011918	0.016966	0.005611	0.002061	0.002061				
14	32444	4	0.04023	0.324627	0.011918	0.016966	0.005611	0.002061	0.002061				
15	32444	4	0.04023	0.324627	0.011918	0.016966	0.005611	0.002061	0.002061				
16	32444	4	0.04023	0.324627	0.011918	0.016966	0.005611	0.002061	0.002061				
17	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
18	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
19	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
20	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
21	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
22	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
23	32450	2	0.04023	0.266055	0.002718	0.000605	0.005773	7.35E-05	7.35E-05				
24	32393	1	0.074713	0.344554	0.002487	0.152862	0.006171	0.018568	0.018568				
25	32393	1	0.074713	0.344554	0.002487	0.152862	0.006171	0.018568	0.018568				
26	32393	1	0.074713	0.344554	0.002487	0.152862	0.006171	0.018568	0.018568				
27	32393	1	0.074713	0.344554	0.002487	0.152862	0.006171	0.018568	0.018568				
28	32393	1	0.074713	0.344554	0.002487	0.152862	0.006171	0.018568	0.018568				
29	32494	4	0.028736	0.3058	0.001467	0.005862	0.003811	0.000712	0.000712				
30	32494	4	0.028736	0.3058	0.001467	0.005862	0.003811	0.000712	0.000712				
31	32494	4	0.028736	0.3058	0.001467	0.005862	0.003811	0.000712	0.000712				
32	32494	4	0.028736	0.3058	0.001467	0.005862	0.003811	0.000712	0.000712				
33	32494	4	0.028736	0.3058	0.001467	0.005862	0.003811	0.000712	0.000712				
Node Level Stats													
Network Statistics													
node_match Summary													

Figure 7: Report generated in excel format.

### Design benefits:

- A user friendly and interactive tool to upload data, perform network statistics and visualize networks. Moreover, the home page of application provides clear instructions on how to upload data and make most of the application.
- Ability to upload excel or csv files containing the network data or attribute data frame and based on that view network metrics and statistics.
- Using statistical models (ERGMs) to gain deeper insights into the network structure with functionalities including network edges, node covariance and node match.
- Moreover, able to view the goodness of fit (GOF) results to check how good the model behaves.
- Allowing users to generate user friendly reports (HTML or excel format), for further analysis of the data.
- In general, allowing users like researchers or data analysts to perform social network analysis efficiently.
- Additionally, a user can even install the application or can view it on mobile devices as well. Application is easy to use and can be accessed using the link: <https://yetanothertechproject.streamlit.app/>

## Project reflection

### Group work reflection:

#### Strategies/Processes that worked:

- Weekly meetings and lab session discussions streamlined the process of creating the web application and helped in communication between the team members.
- Getting constant feedback from the tutor and attending weekly labs to make sure we were on right track.
- Getting the clear idea of the project outcome from the client kept the scope manageable.
- Clear delegation of tasks to make sure everyone is making their valuable contributions and making sure team members were able to give their ideas freely.
- Having Teams meeting and using the chat group created for regular updates ensured that the work is submitted smoothly and according to the rubric.
- Shifting the application framework from Flask application to 'Streamlit' one, proved to be a good choice as it was interactive and facilitated the integration.



- The ERGM integration took time to figure out as Streamlit demanded installation of several packages based on R language, but we were able to achieve that within the given deadline.

### Strategies/Processes that did not worked:

- Shifts in project priorities based on the feedback received accounted for problem in integration of some functions and caused uneven workloads.
- Initially, ALAAMs were supposed to be implemented as well but due to time constraints and change of the project scope, only ERGM was executed.

### Future Improvements:

- Defining the scope of the project before hand and not making major changes without discussions between team members.
- Keeping in check the integration of all functionalities and anticipating the challenges in doing so beforehand so that web application runs smoothly with respect to the expected deliverables.

## Individual work reflection

### Project Tasks

#### *Phase 1*

- As the group was formed, the first task was the research report, and the topics were divided among the team and the topic that was delegated to me was of the statistical model integration in the social network analysis process. Research was done on ERGMs, ALAAMs and SOAMs (Stochastic Actor Oriented Models) leveraging several research papers.
- The research gave a clear understanding as to how the statistical models can be implemented and integrated in a web application effectively.

#### *Phase 2*

- Next, we proceeded with the design concepts of our web application and as a result five designs were proposed and were provided to the tutor.
- Then client feedback was requested following which we decided to work on a 'Streamlit' application and using python as the primary language. 'Streamlit', aimed to streamline the front-end process and facilitate the integration process.
- Next, the integral step was to work on innovation concept, in that all design concepts were documented along with the summary of those, and justification of the design chosen. I proceeded with the latter one and made sure the document was up to the mark.

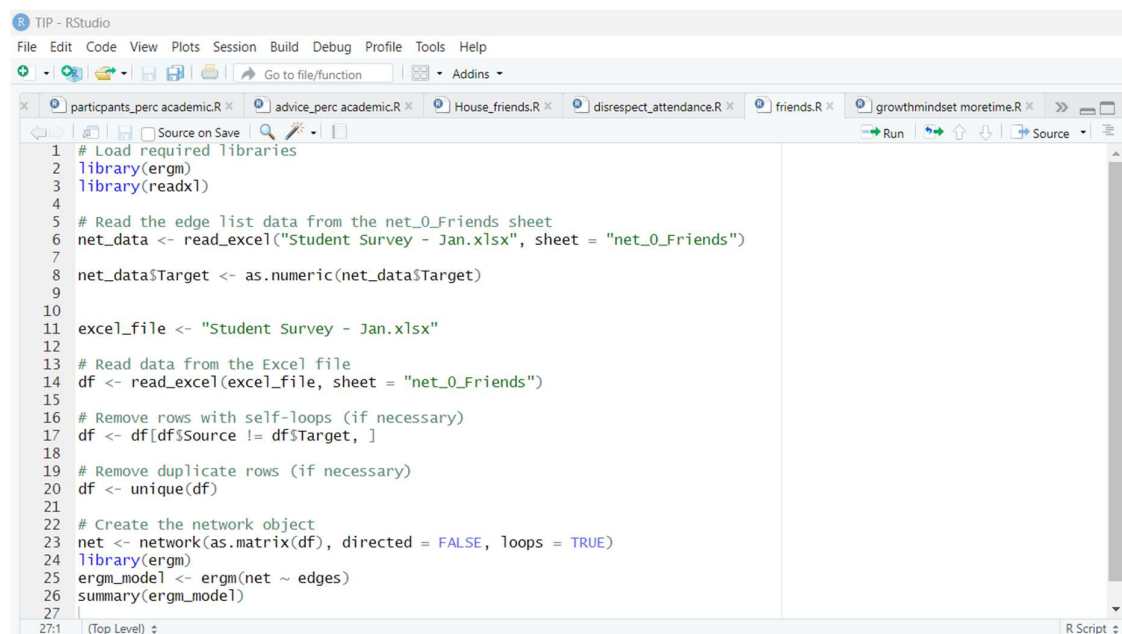
#### *Phase 3*

- Following was the delegation of tasks and deciding the scope of the project where I was delegated to work on implementation of the ERGM model.



ERGM works like a standard logistic regression method giving the probability as whether two nodes are likely to form a tie or not. This model can be used with various types of networks like directed, undirected, valued, unvalued, and bipartite. It differs from the standard regression method as it can be used for simulation of similar networks, they allow interactions between individual attributes at dyadic level (interaction of nodes within the network), edge attributes and predictor networks.[5]

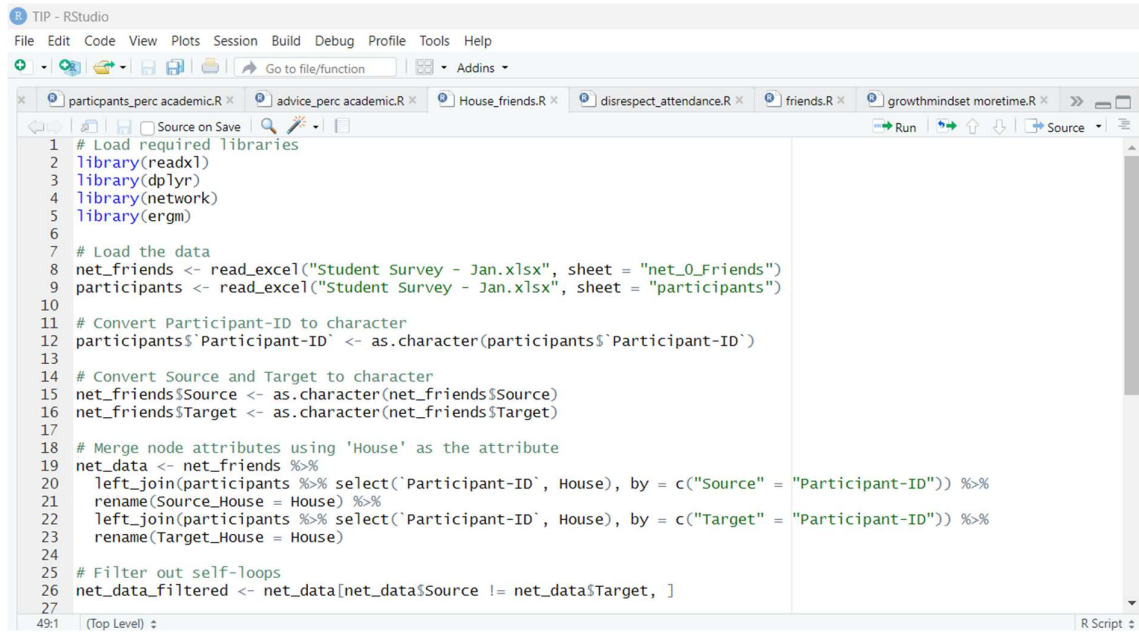
- At first because of limited knowledge of python packages which will be helpful to implement ERGM functionality, I decided to proceed with R language. As it has all the necessary packages associated with the networks and ERGM model in general and python does not directly have the functionality to do so.
- Proceeding with that I started by working in RStudio, which is an integrated development environment for R. As the client already provided us with data to work on, I implemented ERGM using that.

The image is a screenshot of the RStudio integrated development environment. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and running code. The main window displays an R script with the following code:

```
1 # Load required libraries
2 library(ergm)
3 library(readxl)
4
5 # Read the edge list data from the net_0_Friends sheet
6 net_data <- read_excel("Student Survey - Jan.xlsx", sheet = "net_0_Friends")
7
8 net_data$Target <- as.numeric(net_data$Target)
9
10
11 excel_file <- "Student Survey - Jan.xlsx"
12
13 # Read data from the Excel file
14 df <- read_excel(excel_file, sheet = "net_0_Friends")
15
16 # Remove rows with self-loops (if necessary)
17 df <- df[df$Source != df$Target, ]
18
19 # Remove duplicate rows (if necessary)
20 df <- unique(df)
21
22 # Create the network object
23 net <- network(as.matrix(df), directed = FALSE, loops = TRUE)
24 library(ergm)
25 ergm_model <- ergm(net ~ edges)
26 summary(ergm_model)
27
```

The status bar at the bottom indicates the current line is 27:1 and the file is (Top Level). The right side of the window is empty, showing the Plots pane.

Figure 8: Implementation of ERGM(Bernoulli) with R

The image is a screenshot of the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and a search bar. The main editor window displays an R script with the following code:

```
1 # Load required libraries
2 library(readxl)
3 library(dplyr)
4 library(network)
5 library(ergm)
6
7 # Load the data
8 net_friends <- read_excel("Student Survey - Jan.xlsx", sheet = "net_0_Friends")
9 participants <- read_excel("Student Survey - Jan.xlsx", sheet = "participants")
10
11 # Convert Participant-ID to character
12 participants$`Participant-ID` <- as.character(participants$`Participant-ID`)
13
14 # Convert Source and Target to character
15 net_friends$Source <- as.character(net_friends$Source)
16 net_friends$Target <- as.character(net_friends$Target)
17
18 # Merge node attributes using 'House' as the attribute
19 net_data <- net_friends %>%
20   left_join(participants %>% select(`Participant-ID`, House), by = c("Source" = "Participant-ID")) %>%
21   rename(Source_House = House) %>%
22   left_join(participants %>% select(`Participant-ID`, House), by = c("Target" = "Participant-ID")) %>%
23   rename(Target_House = House)
24
25 # Filter out self-loops
26 net_data_filtered <- net_data[net_data$Source != net_data$Target, ]
27
```

The status bar at the bottom indicates the script is at line 49, column 1, and is at the top level.

Figure 9: Implementation of ERGM (node match) with R

- After more refined research I implemented the same in python using ‘rpy2’ module, which helps python interface to work with R language. Later, as we wanted flexible ERGM functionality, with the help of my team member, the function no longer required providing data manually.

```

friends edges.py > ...
1  import pandas as pd
2  import rpy2.robjects as ro
3  from rpy2.robjects.packages import importr, isinstalled
4  from rpy2.robjects import pandas2ri
5  from rpy2.robjects.conversion import localconverter
6
7  # Check and handle package imports
8  network = importr('network', on_conflict='warn') if isinstalled('network') else None
9  ergm = importr('ergm', on_conflict='warn') if isinstalled('ergm') else None
10
11 # Load data using pandas
12 net_data = pd.read_excel("Student Survey - Jan.xlsx", sheet_name="net_0_Friends")
13 net_data = net_data[net_data['Source'] != net_data['Target']].drop_duplicates()
14
15 # Aggregate parallel edges if any (counting the occurrences of each edge)
16 net_data['weight'] = 1 # Assign a weight of 1 to each edge initially
17 net_data = net_data.groupby(['Source', 'Target']).weight.sum().reset_index()
18
19 # Ensure pandas2ri is activated for the session
20 pandas2ri.activate()
21
22 # Convert pandas DataFrame to R DataFrame
23 with localconverter(ro.default_converter + pandas2ri.converter):
24     r_net_data = ro.conversion.py2rpy(net_data)
25
26 # Assign the R DataFrame to a variable in R's global environment
27 ro.globalenv['df'] = r_net_data
28
29 # Execute R code for network analysis
30 ro.r('''
31 df$Source <- as.numeric(df$Source)
32 df$Target <- as.numeric(df$Target)
33 # Set multiple=TRUE to handle parallel edges if needed
34 net <- network::network(df, directed = FALSE, loops = TRUE, multiple = TRUE)
35 ergm_model <- ergm::ergm(net ~ edges)
36 summary_ergm <- summary(ergm_model)
37 ''')

```

Figure 10: Implementation of ERGM with python and R(rpy2)

```

ergmforapp.py > perform_ergm_analysis
1 import rpy2.robj as ro
2 from rpy2.robj import pandas2ri
3 from rpy2.robj.conversion import localconverter
4
5 def perform_ergm_analysis(network_df, attribute_df, selected_attribute, edges_only=False):
6     output_file_path="ergm_analysis_results.txt"
7     status = 'Error Occured'
8     if edges_only:
9
10         pandas2ri.activate()
11         with localconverter(ro.default_converter + pandas2ri.converter):
12             r_net_data = ro.conversion.py2rpy(network_df)
13
14             ro.globalenv['df'] = r_net_data
15
16         try:
17             ro.r(f'''
18                 library(network)
19                 library(ergm)
20                 df$Source <- as.character(df$source)
21                 df$Target <- as.character(df$target)
22                 net <- network::network(df, directed = TRUE, loops = FALSE)
23
24                 # ERGM formula for edges only
25                 formula <- "net ~ edges"
26                 ergm_model <- ergm::ergm(as.formula(formula))
27                 summary_ergm <- summary(ergm_model)
28                 writelines(capture.output(summary_ergm), "{output_file_path}")
29                 ''')
30             status = f"ERGM analysis completed. Results saved to {output_file_path}"
31
32         except Exception as e:
33             status = f"An error occurred: {e}"
34     else:
35         attribute_df = attribute_df[['NodeID', selected_attribute]]
36         attribute_df.dropna(subset=[selected_attribute], inplace=True)

```

Figure 11: Implementation of ERGM with respect to the application.

```

attribute 1.txt
1 Call:
2 ergm::ergm(formula = net ~ edges + nodematch("House", diff = FALSE))
3
4 Maximum Likelihood Results:
5
6      Estimate Std. Error MCMC % z value Pr(>|z|)
7 edges      -3.61003    0.03909      0  -92.34  <1e-04 ***
8 nodematch.House  1.39786    0.06170      0   22.66  <1e-04 ***
9 ---
10 Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
11
12 Null Deviance: 42213 on 30450 degrees of freedom
13 Residual Deviance: 9394 on 30448 degrees of freedom
14
15 AIC: 9398 BIC: 9415 (Smaller is better. MC Std. Err. = 0)
16

```

Figure 12: A typical result of ERGM functionality

- Further, modifications were made to the code, and a uniform function was written implementing ERGM functionality. The crucial part was to have choice for three operations based on the data provided, I opted for network edges

(Bernoulli), node covariance and node match. Here, 'edges' is for connection between pair of nodes, negative coefficient of which indicates a sparse network. 'Node covariance' indicated how individual attributes affect tie formation. 'Node match' indicates the homophily that whether the nodes sharing same attribute are likely to form connections or not.

#### Phase 4

- As per the feedback provided from the client on the project progress, implementation of 'goodness of fit'(GOF) was demanded. GOF states how a model's predicted network statistics match the observed network statistics for the actual network.
- Following that the GOF was implemented and was integrated into the web application which works for 'edges' data or data where attributes are mentioned. Outdegree and indegree parameters were checked on. Here, outdegree means the number of outgoing connections a node has and indegree means the number of incoming connections a node receives.

```
C: > Users > DELL > Desktop > Stats > gofinstrapp.py > perform_gof_analysis
1  def perform_gof_analysis(ergm_model, gof_output_file_path):
2      try:
3          # Assuming the ERGM model is available in the R environment as ergm_model
4          ro.r(f'''
5              library(ergm)
6              gof_results <- gof(ergm_model, GOF=~odegree+idegree)
7              writeLines(capture.output(gof_results), "{gof_output_file_path}")
8              ''')
9      except Exception as e:
10         print(f"An error occurred during GoF analysis: {e}")
11
```

Figure 13: Implementation of goodness of fit for the application.

## Goodness of Fit Results

Out Degree Goodness of Fit

	degree	obs	min	mean	max	p-value
0	odegree0	13	0	0.15	1	0.00
1	odegree1	5	0	1.50	5	0.04
2	odegree2	13	0	4.86	13	0.06
3	odegree3	17	4	11.15	20	0.08
4	odegree4	10	9	18.35	31	0.02
5	odegree5	25	16	25.30	33	1.00
6	odegree6	16	14	27.58	43	0.02
7	odegree7	22	17	26.62	37	0.48
8	odegree8	14	13	21.43	30	0.08
9	odegree9	6	7	15.42	25	0.00
10	odegree10	6	5	10.36	17	0.14
11	odegree11	6	0	5.87	13	1.00
12	odegree12	3	0	3.31	10	1.00
13	odegree13	0	0	1.75	5	0.38
14	odegree14	4	0	0.83	4	0.02

Figure 14: Out degree result for goodness of fit

In Degree Goodness of Fit

	degree	obs	min	mean	max	p-value
0	idegree0	2	0	0.21	3	0.08
1	idegree1	9	0	1.46	5	0.00
2	idegree2	14	0	5.20	12	0.00
3	idegree3	17	3	11.27	22	0.12
4	idegree4	17	7	18.51	29	0.88
5	idegree5	11	13	24.91	36	0.00
6	idegree6	24	14	27.31	38	0.52
7	idegree7	19	15	25.87	37	0.16
8	idegree8	12	12	21.42	32	0.02
9	idegree9	8	6	16.06	26	0.10
10	idegree10	14	3	10.11	17	0.32
11	idegree11	7	2	6.23	12	0.88
12	idegree12	12	0	3.50	8	0.00
13	idegree13	1	0	1.69	5	0.94
14	idegree14	2	0	0.73	4	0.34
15	idegree15	0	0	0.24	2	1.00

Figure 15: In degree results for goodness of fit.

Network Goodness of Fit

	model statistic	obs	min	mean	max	p-value
0	edges	1159	1095	1156.93	1233	0.94
1	nodematch.Perc_Effort	363	321	365.28	414	0.98

Figure 16: General network goodness of fit results

- Next, the application was tested, and the model was validated along with GOF results. Lastly, we reached the final project demonstration along with a presentation where the client and our tutor saw the final product.

## Conclusion and Recommendations

- The main achievement was to successfully implement a statistical model called ERGM and goodness of fit (GOF) for the same. The statistical model integration was a new concept and although it took effort and research, it enhanced and gave us deeper understanding of the network and its structure.
- Further, in the future iterations the model could be more enhanced handling different types of data, have more functions other than Bernoulli's, node covariance or node match to give more enhanced statistics on social networks. This requires additional time and research.
- Moreover, in future iterations of the projects, I would like the SAOM model to be implemented. This model is used for analyzing social interactions and behavior with respect to time.

## References

1. Streamlit • The fastest way to build and share data apps. (n.d.). Streamlit.io. <https://streamlit.io/>
2. NetworkX. (n.d.). NetworkX — NetworkX documentation. Networkx.org. <https://networkx.org/>
3. Introduction — pyvis 0.1.3.1 documentation. (n.d.). Pyvis.readthedocs.io. <https://pyvis.readthedocs.io/en/latest/introduction.html>
4. Introduction to ERGMs. (n.d.). Social Network Analysis for Anthropologists. <https://eehh-stanford.github.io/SNA-workshop/ergm-intro.html>
5. Introduction to ERGMs. (n.d.). Social Network Analysis for Anthropologists. <https://eehh-stanford.github.io/SNA-workshop/ergm-intro.html>