Research Project:

Technical Documentation

Dalton Murray

College of Business & I.T., Lawrence Technological University

INT 6303, Introduction to Social Media Data Analytics

Dr. Mazin Al Hamando

July 18, 2023

**Introduction**

This document serves as the technical documentation for my overall research project, paper combined, and will work on a step-by-step basis.

**Step 1 – Redefining project**

My initial problem was as follows:

Computer hardware is a rapidly changing world, with these changes there is constant releases of new hardware and new requirements to be able to do basic things such as accessing applications and running games. These new requirements require newer hardware which has an incredibly high performance, as a result of this there is an ever-growing price to this new hardware. The performance to cost ratio is becoming astonishing, making it less worth the money to buy the new hardware but it is becoming necessary to purchase because of the requirements of applications and games. There is also a growing monopoly which further allows companies such as Nvidia not to worry about their prices because there are limited options available.

However, I want to redefine this and be more specific. Due to limitations of time, instead of targeting all new hardware and requirements with price to performance I will be targeting specifically graphics cards. I have also changed my dataset from large to medium, although this depends on perspective. This is also because of Reddit API actions/hour limits, thankfully I did not run into the issue with the new API changes of having to pay. I have already collected my data and have ended up with 8 primary posts and 2,365 comments. I am also specifying that I will not be filtering comments or performing sanitization. If I had more time in order to get more accurate results I would perform full filtering and sanitization, however, with this many comments it would be difficult to do so. I did, however, in the end perform limited filtering. I removed moderator comments/warnings from the comments as they were not on topic of the posts nor provided anything for discussion.

Due to other limitations, not specifically time, but I will also not be performing sentiment analysis. I originally hoped to do so, however, I determined that sentiment analysis would not provide much for my analysis or end-results. Sentiment analysis would be able to provide some information, however, there is not any publicly trained sentiment analysis models on Reddit which I could find, and if I did not use a Reddit specific one the results would be swayed as I have tested as it removed Reddit specific terminology and was more focused on non-comment based things such as full paragraphs and sentences and did not take into account many other factors. In a future project I would like to create my own model for sentiment analysis and my own program to determine sentiment based on individual words and multi-topic single comments which talks about many brands and is smart enough to determine what words are for what brand which I couldn’t find any example of currently for Reddit.

The project portion will also be limited to only data from Reddit, and the actual paper will have combined data from multiple sources including scholarly sources.

**Step 2 – Data collection**

In order to properly collect data, I needed to find posts which were primarily on-topic of GPU pricing and performance, which I was able to do by performing a few Google searches and Reddit searches to find posts. Next, this is the most troubling part, I had to decide if I want to create a Reddit scraper or use an existing one. I determined that I did not have enough time to create my own and would search for an existing one. One of my primary criteria was that it be free to use and have practically no limit. I later had to add to my criteria that it be in a correct raw format, which was tricky. I eventually found one on GitHub called URS, Universal Reddit Scraper. Unfortunately, however, the documentation was quite poor and lacking, but I managed to get it working correctly. Due to new limitations on Reddit, I am actually quite lucky that I collected the data in-time as they are now charging a ridiculous amount of money for API usage, and I got my data scraped just in-time.

To start, I had to install a large amount of different packages and frameworks to make URS work, I also needed to get Reddit API access which at the time could be easily done through going to Reddit Apps, or going to this link: <https://www.reddit.com/prefs/apps>. I created an app, got my id, secret key, and configured the redirect URL. After this you need to properly download the GitHub repo and set it up. I installed poetry, another specific Python version, and a few other modules which I had to import but wouldn’t do automatically. I also had to set an environment variable for PYTHONPATH. Following this, I finally figured out how to properly run URS by running the command `poetry run python Urs.py –check`. This would tell me how many actions on the API I am still allowed to take. I found the command after a while to scrape individual posts, which was `poetry run Python Urs.py -c \*link\* –raw`. I had to specify `--raw` in order to get the proper formatting which is needed in R so that the links would work correctly otherwise URS wanted to format it in a different way.

**Step 3 – Limited data cleanup**

I got my data but had to perform some corrections. The data given to me was in .json format which will need work. I found an easy online .json to .csv converter (<https://codebeautify.org/json-to-csv>). I then converted these .csvs to .xlsx so I can modify it more easily in Excel.

I moved the `parent\_id` column to a and moved the `id` column to b. I however renamed the `id` column to `comment\_id` to specify further down this is the ID of the comment. I also had to edit the `parent\_id` column by removing the prefixes as they were added in by URS and would not work with R to specify correctly connections, and I did this by an easy formula with `=RIGHT(B2, LEN(B2) - 3)`which removed the first 3 characters of the value, I then set the corrected id’s to the actual value of the formula rather than keeping the formula in its place by copying them all then pasting them all as the value and not the formula. I also removed the `body\_html` column as it was unnecessary as I already have the comments and only put the comments in HTML formatting which I did not want. I also removed the `distinguished` column as it was for moderator posts which I removed previously. I also removed the `stickied` column as no comments were stickied to the top of the posts. I also removed `link\_id` column and replaced it with the main post name for simplicity. I also had to separate this spreadsheet into two spreadsheets due to limitations in R. In one spreadsheet I had the `parent\_id` column and the `comment\_id` column. The other spreadsheet I had to rearrange and add some extra information. I have an `ID` column, an `author` column, a `post` column which doubles as the comment column, a `score` column for karma, a `created\_utc` column, an `edited` column, a `is\_submitter` column, a `post` column, a `parent\_id\_old` column, a `comments` column which are for the primary posts, a `original\_comment` column also for the primary posts and not comments. After this I was essentially done with the data collection and cleanup parts and thought I could work on the analysis steps.

**Step 4 – The issue**

Before I could start analysis steps, for quite a long time after cleaning up the data I had an issue. I was trying to create a simple plot in R with my data, but it would never work. I already converted the .xlsx files to .csv files, but I kept getting an error “Some vertex names in edge list are not listed in vertex data frame.” I just couldn’t figure out this issue, I tried remaking all of my data and redoing my R file. I spent over two weeks trying to figure this out, but eventually did it. As it turns out, using Excel’s built in file converter from .xlsx to .csv for some reason R did not like. I manually recreated the .csv files from the .xlsx files, and finally R took them!

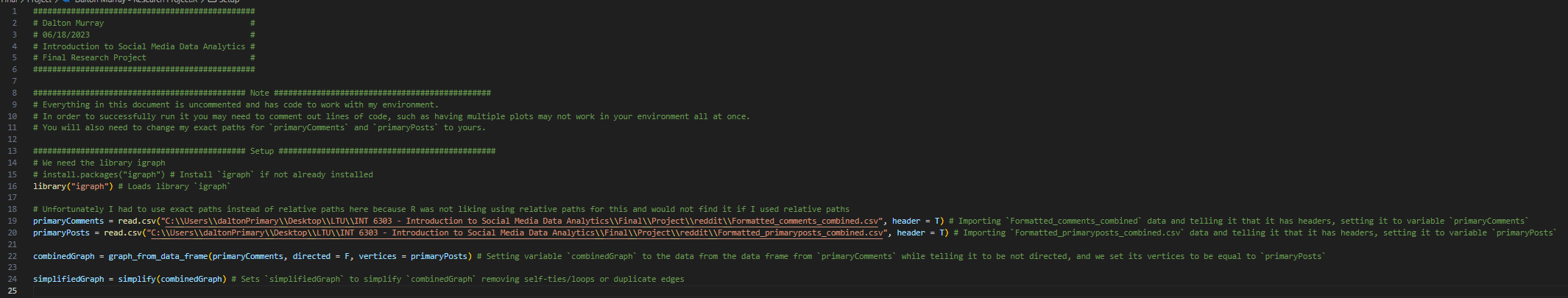
**Step 5 – Standard functions and analysis**

Within this step I wrote all of the code in my R file. I felt that it would be a good idea to take everything which we have learned within the tutorials and use it as a foundation to build off of. To do this, I took my code from my previous tutorial files and put it in my R file for this project. I then modified it to work with this project as well as organized it better. I then improved upon the code further. This step I will include a picture of the code and then a picture of the output. This will be considered the large part of my data analysis for the project; however, I have also performed manual data analysis by reading the data and looking for trends myself. To make this document more logical I have separated standard functions, which will be in this step, and graphs and plots which will be in the next step.

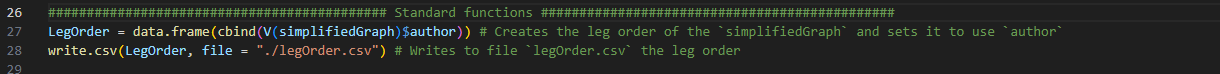
\*Please note that all functions and plots/graphs will be ran on my personal server. The server core components are: AMD Ryzen Threadripper Pro 5996WX CPU, Asus Strix RTX 4090 OC, 1TB of DDR4 3200 ram.

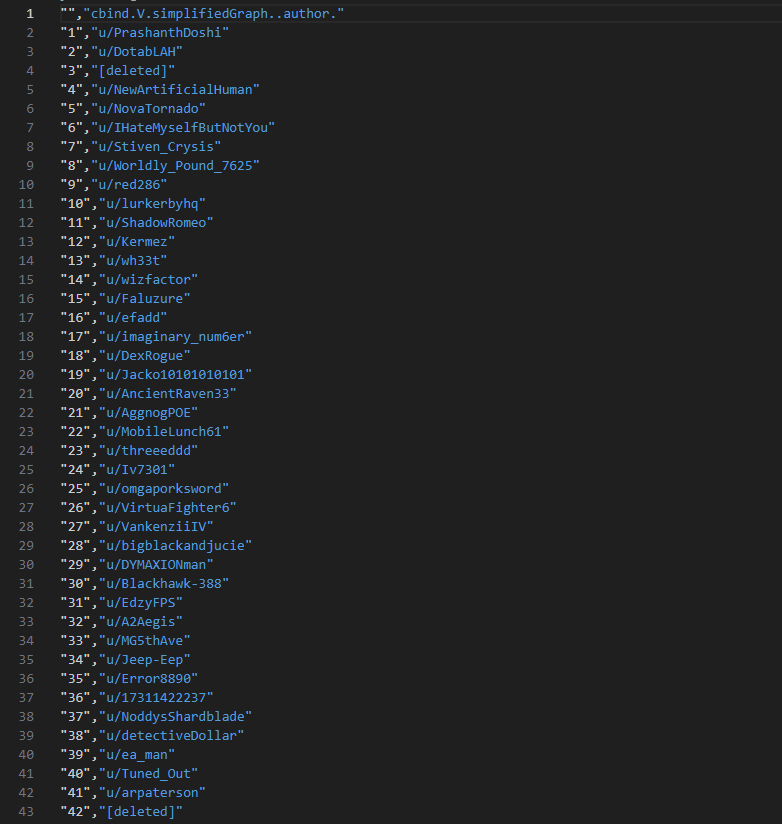
There were some issues with getting R to run some things/it got hung up on and refused to fully load, specifically at the Plots & Graphs section.\*

This will be the basic functions section.

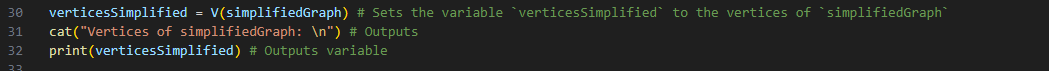


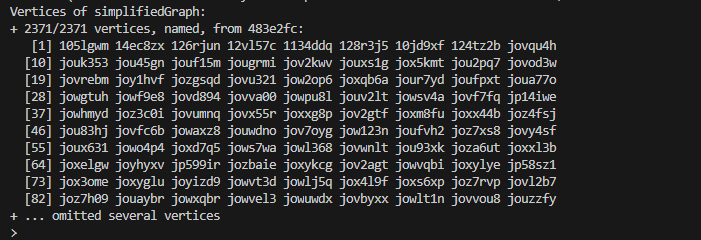
\*No output, this is just for setting up everything to be used later on.

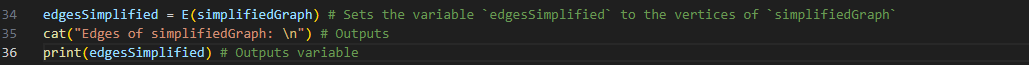


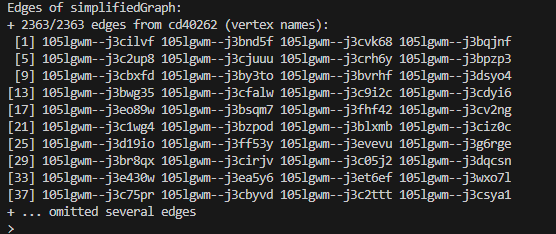


\*Goes to 2372 lines







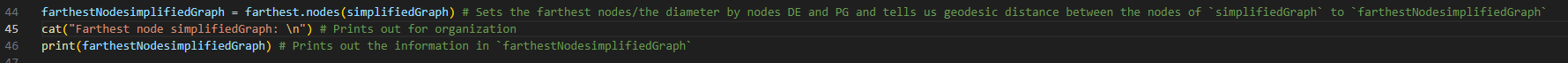


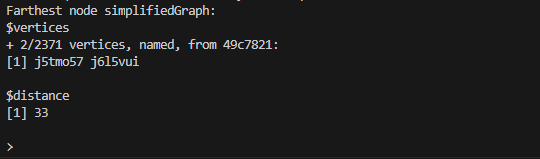


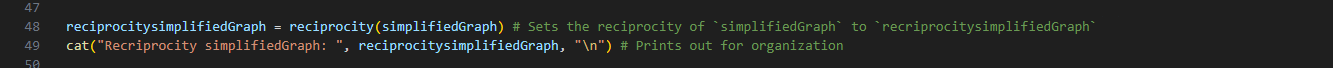
****

****





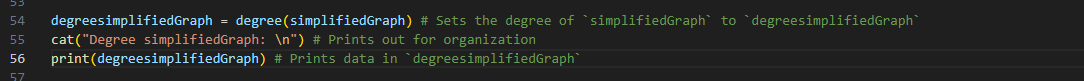


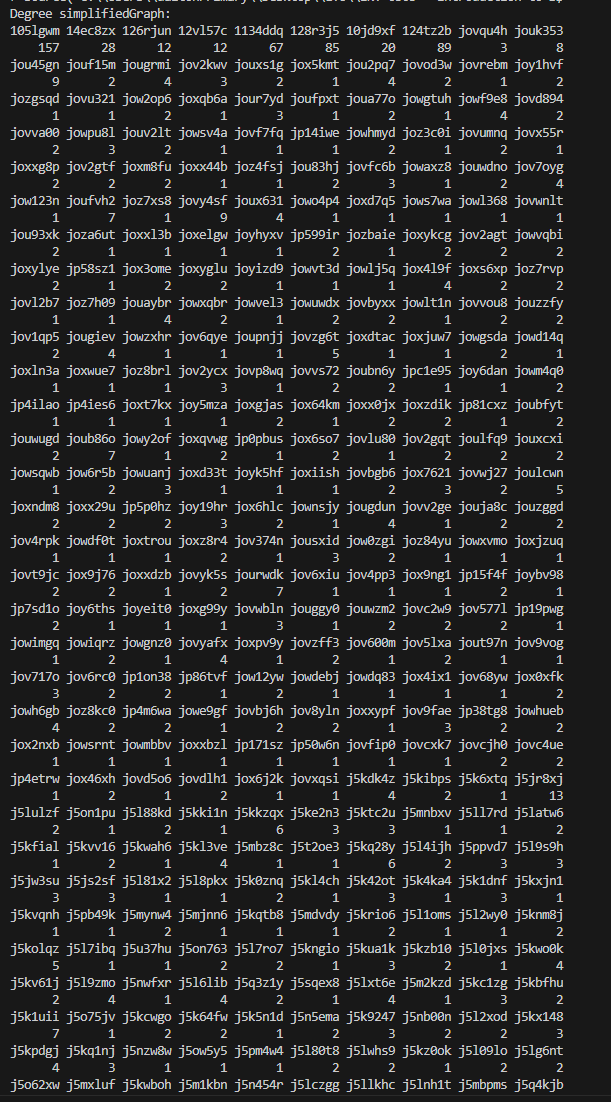




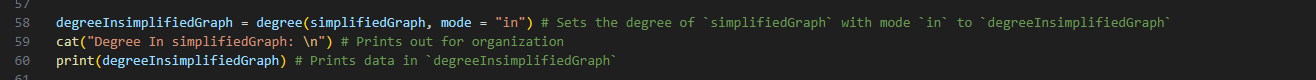


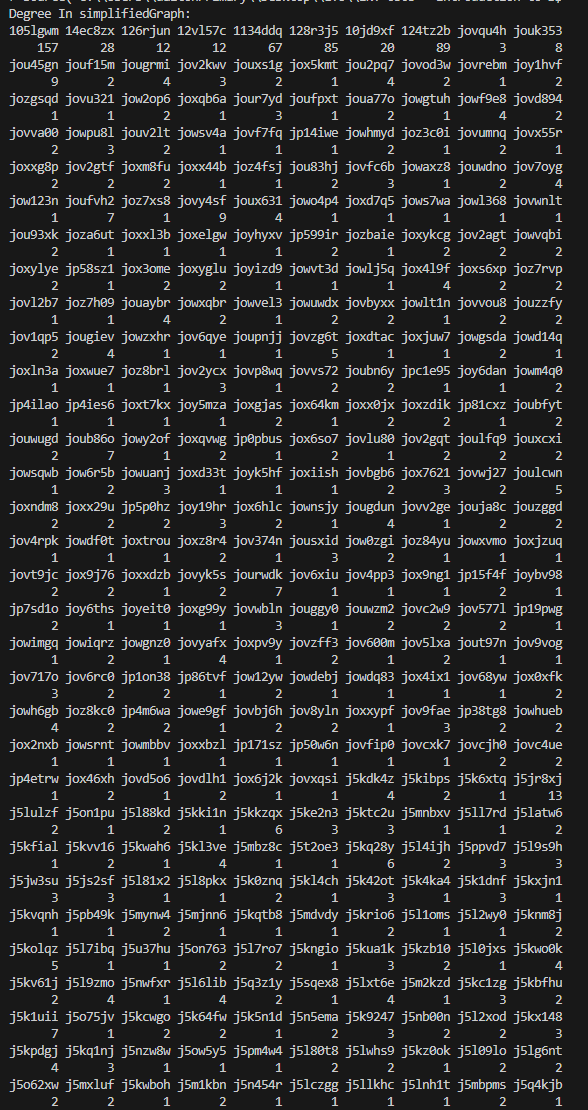




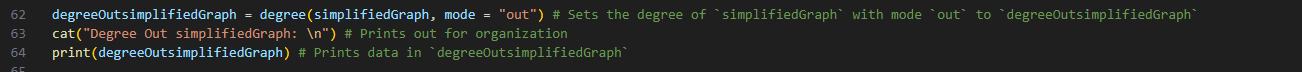


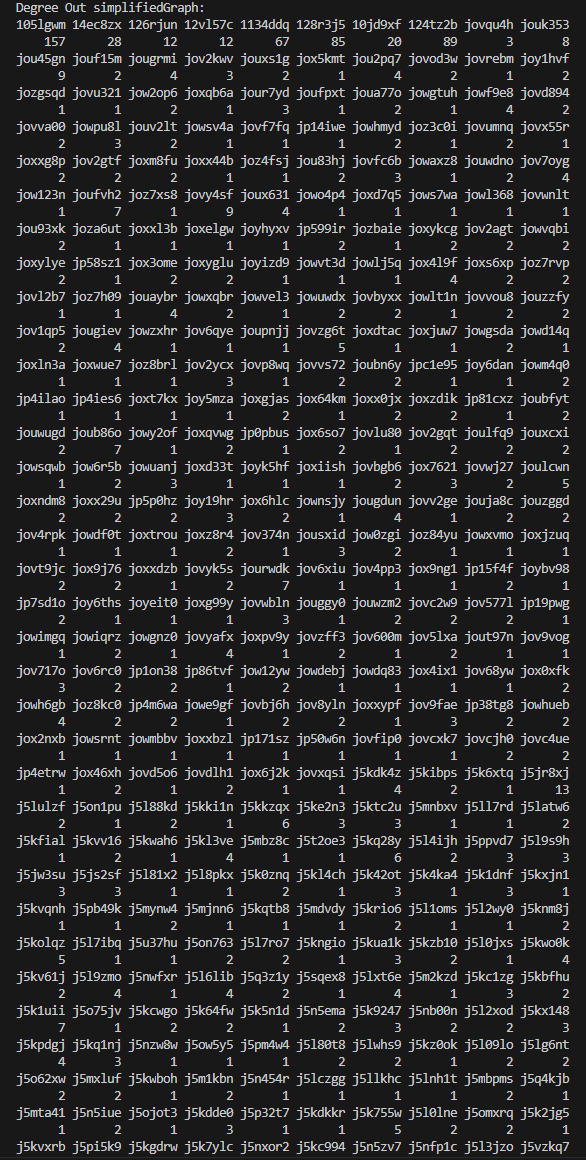
\*Many more lines



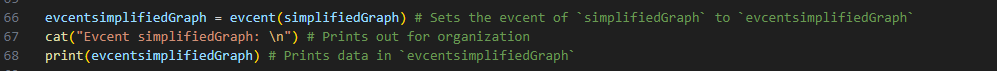


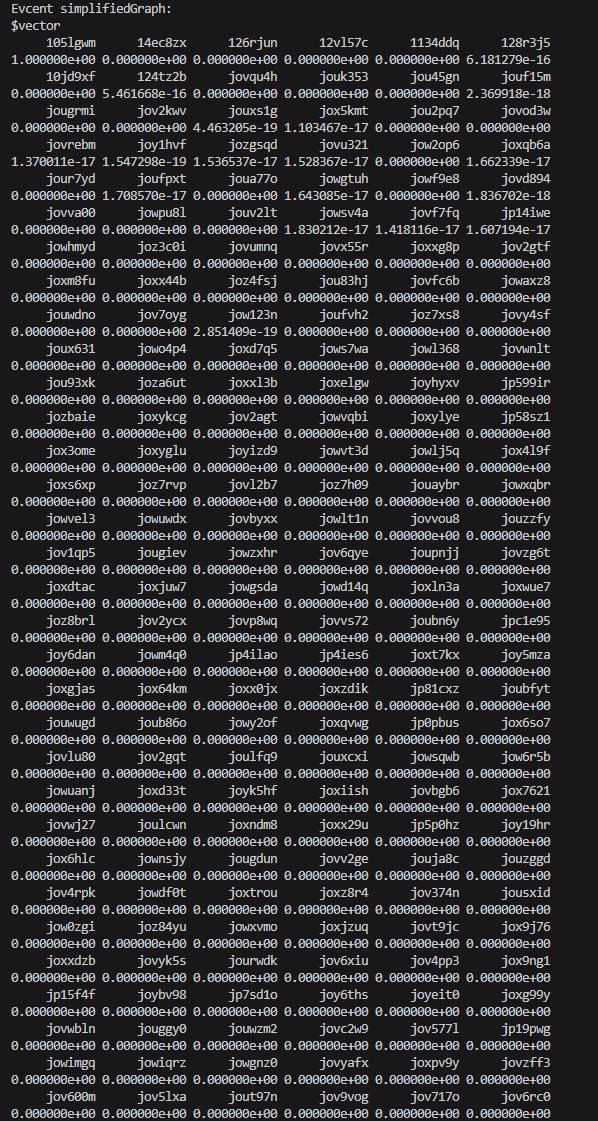
\*Many more lines



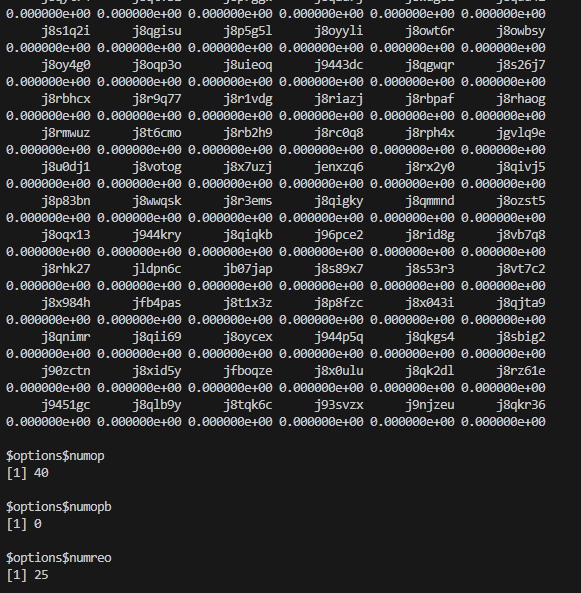


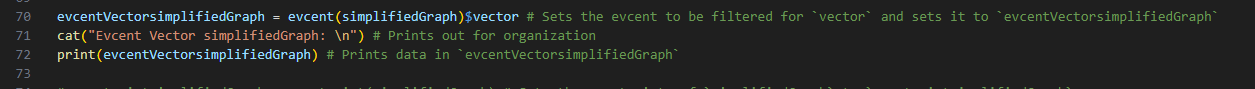
\*Many more lines

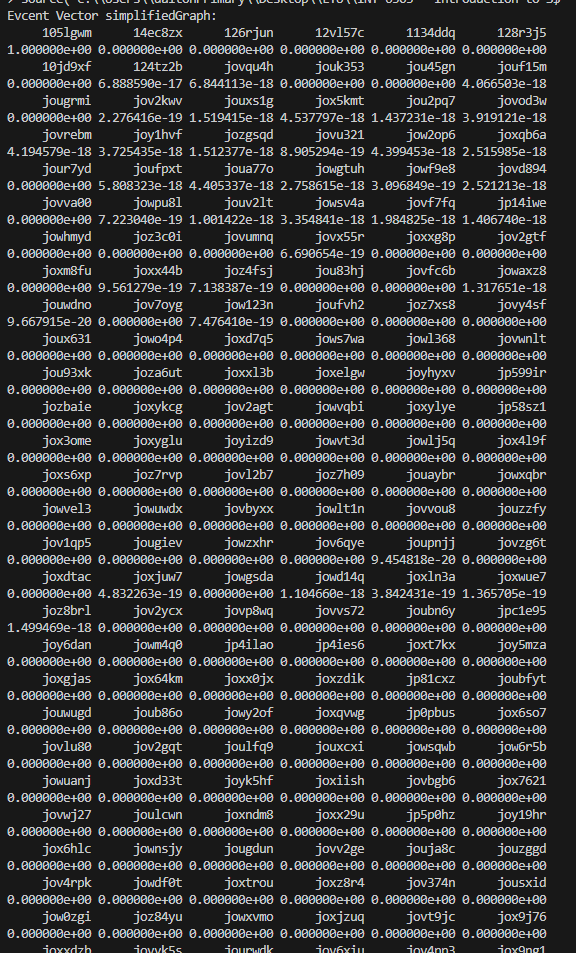




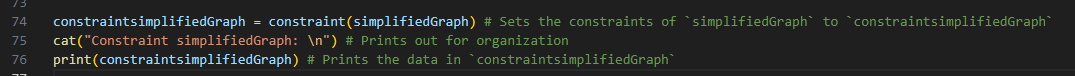
\*Many more lines

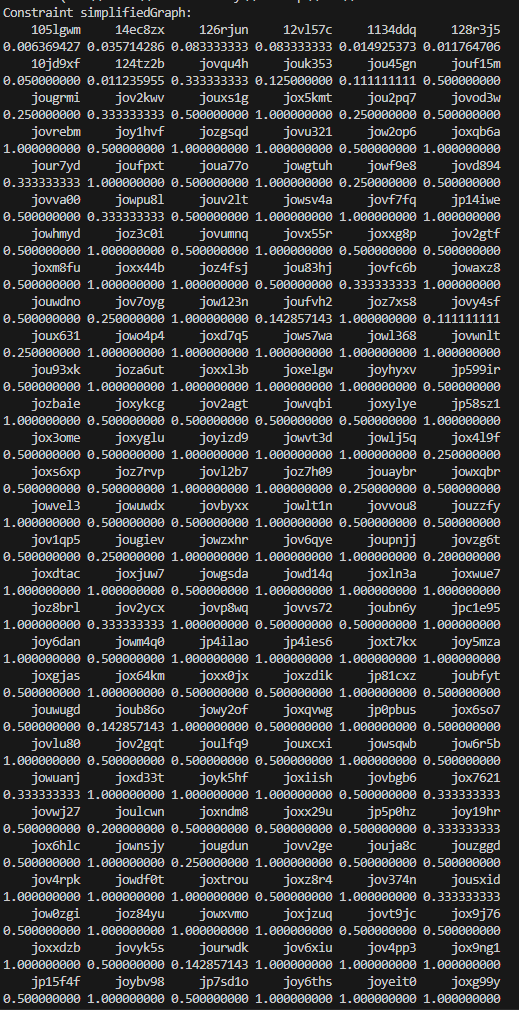




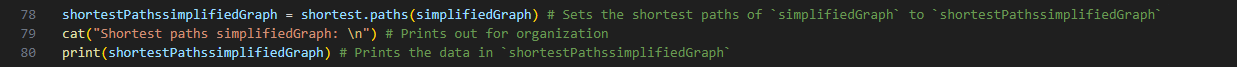


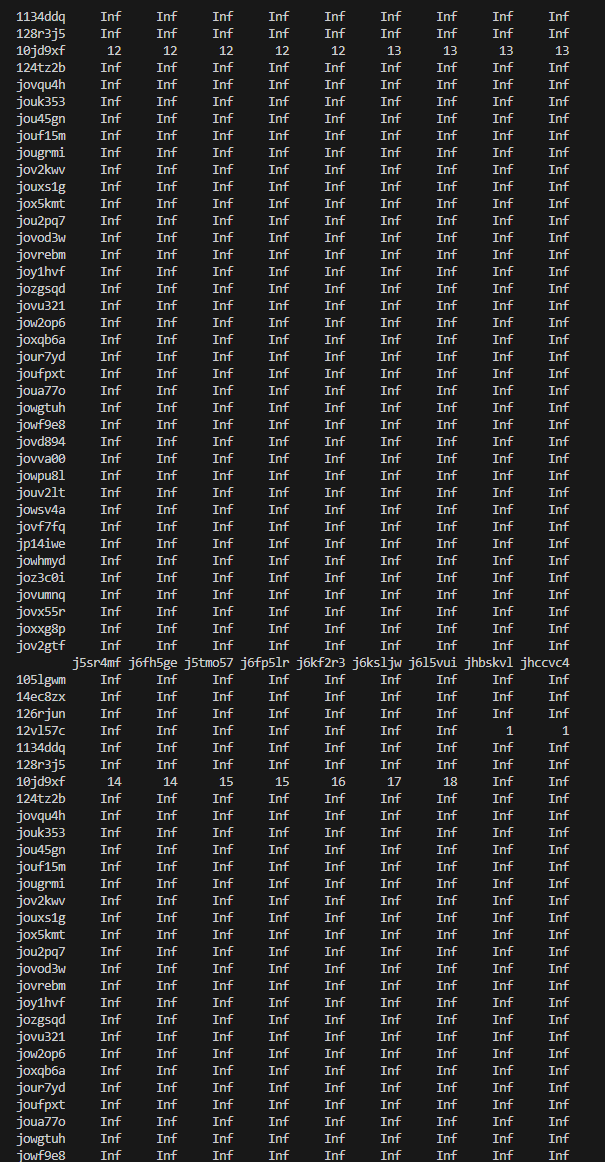
\*Many more lines



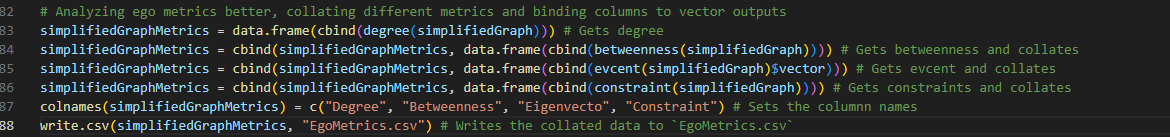


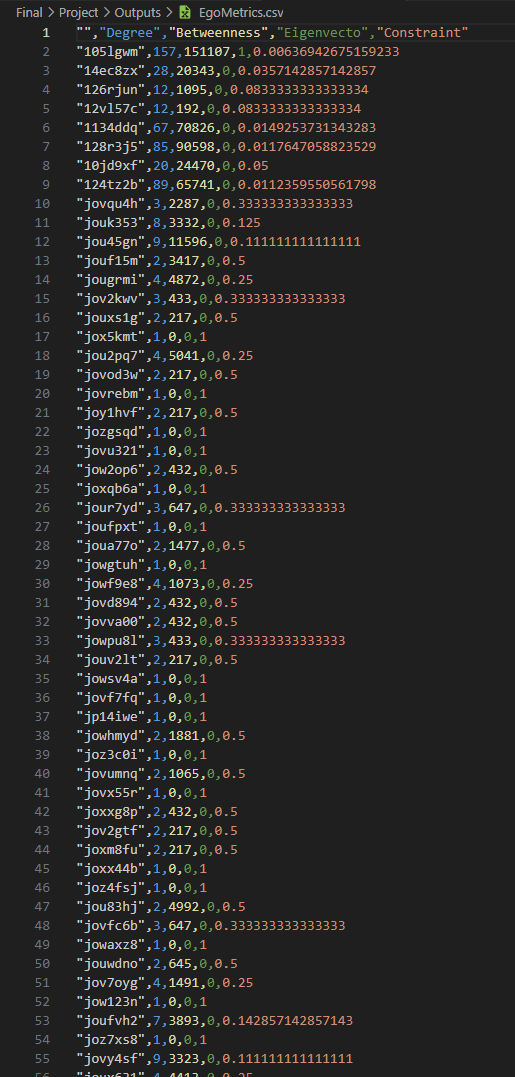
\*Many more lines





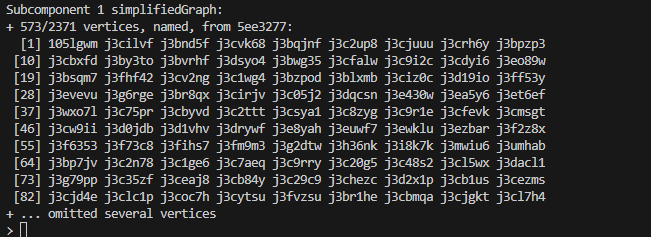
\*Many more lines

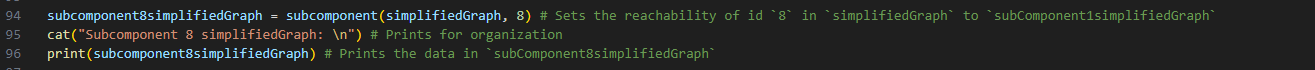


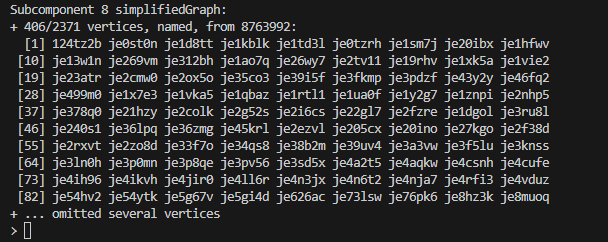


\*Many more lines





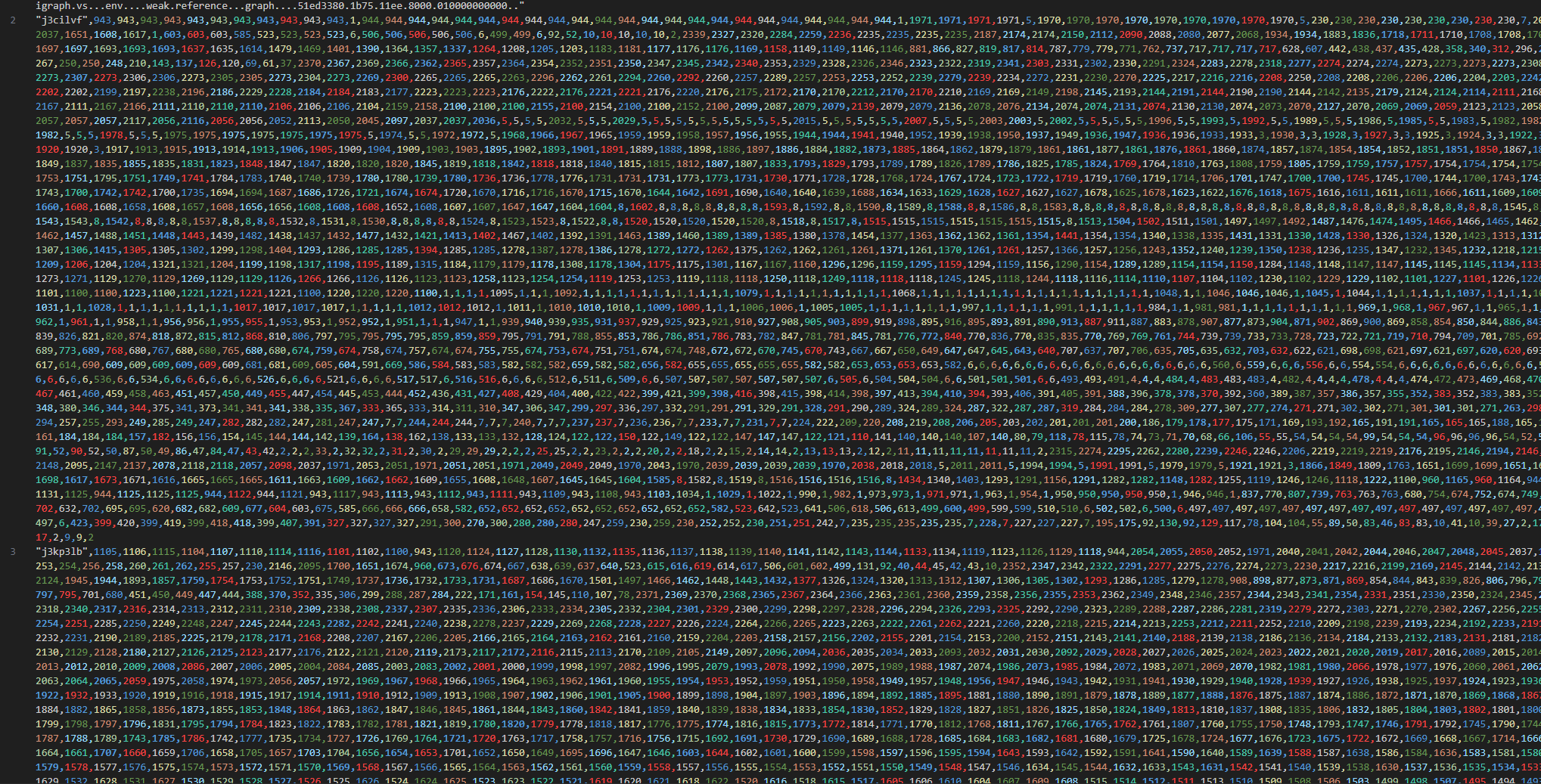






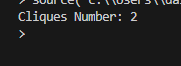


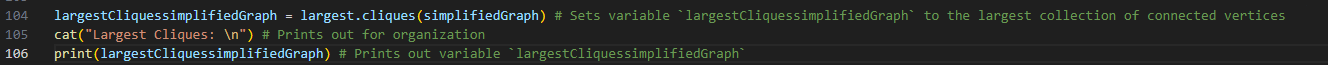
\*Many more lines

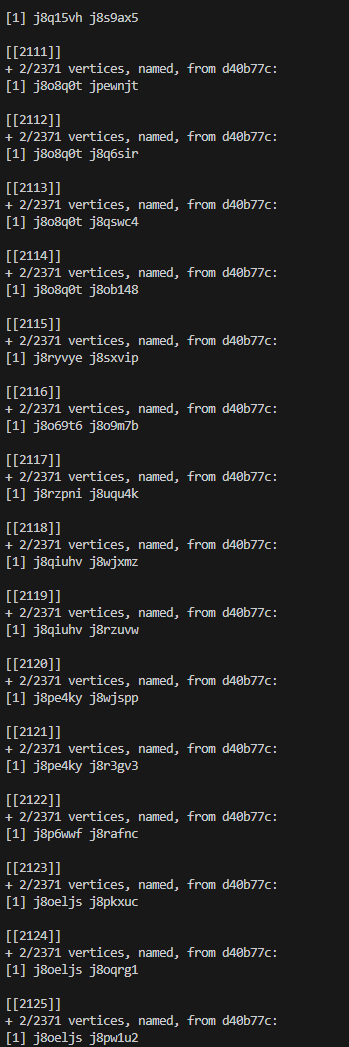


\*Many more lines









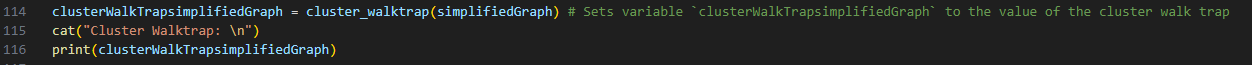
\*Many more lines

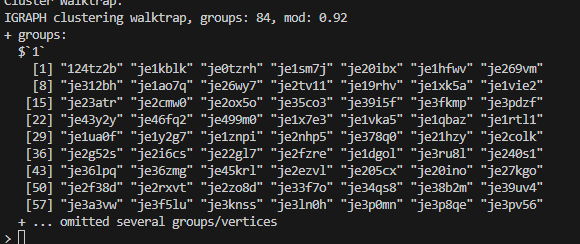


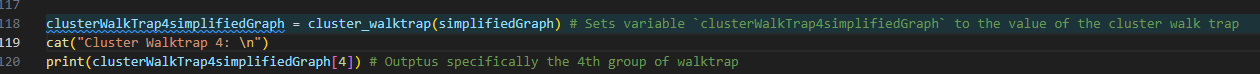


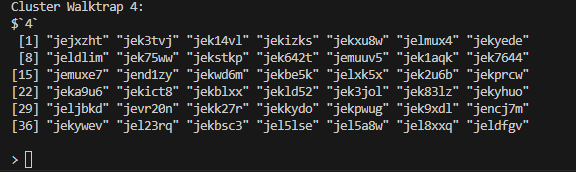




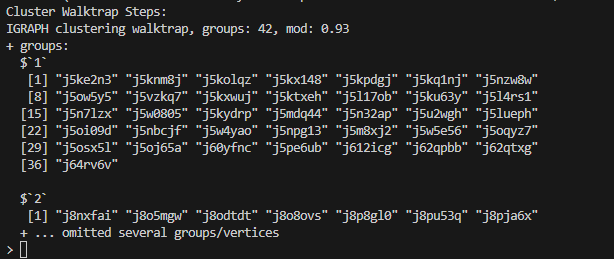


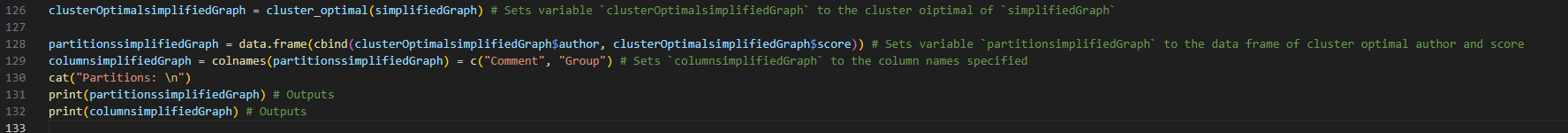








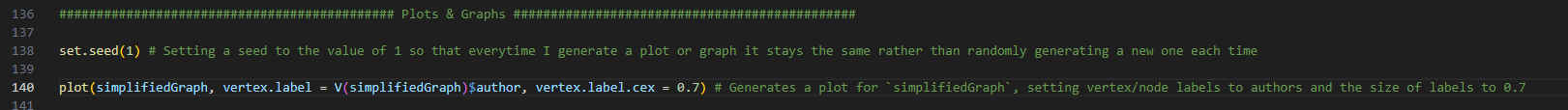


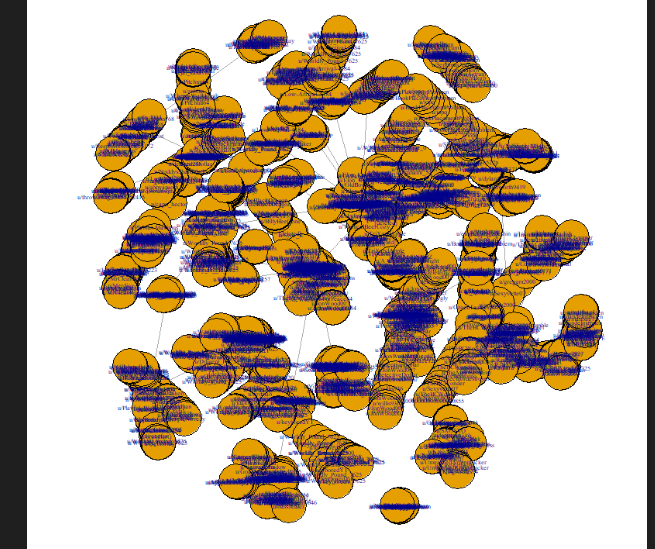


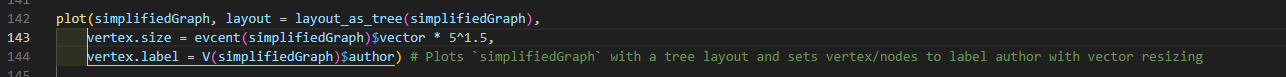
\*Refused to run on server

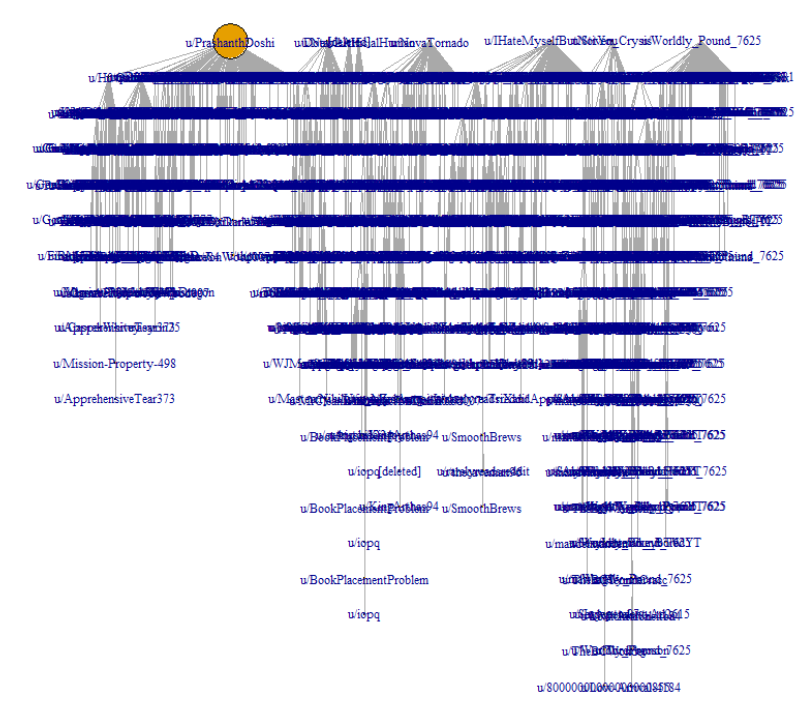
This section will be about plots and graphs.

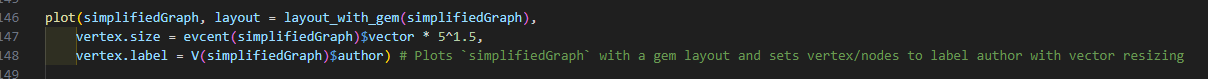
Please note that there is an incredible amount of overlap in nodes with most plots.



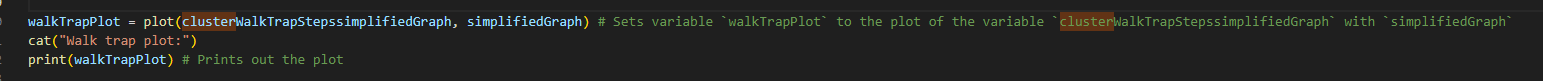


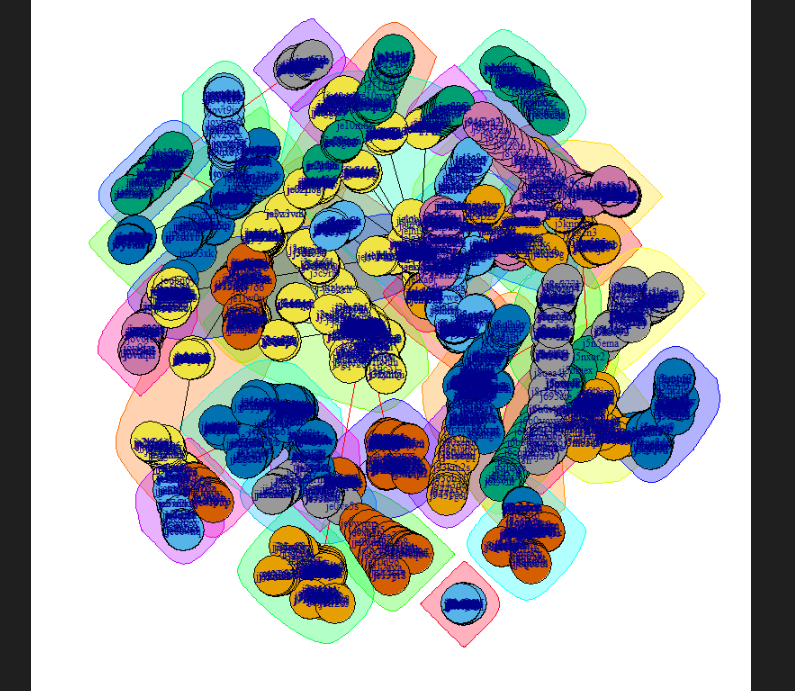


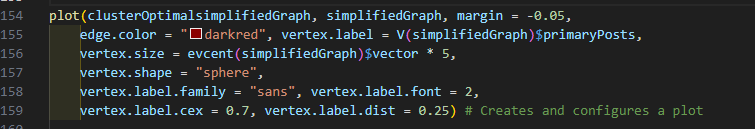




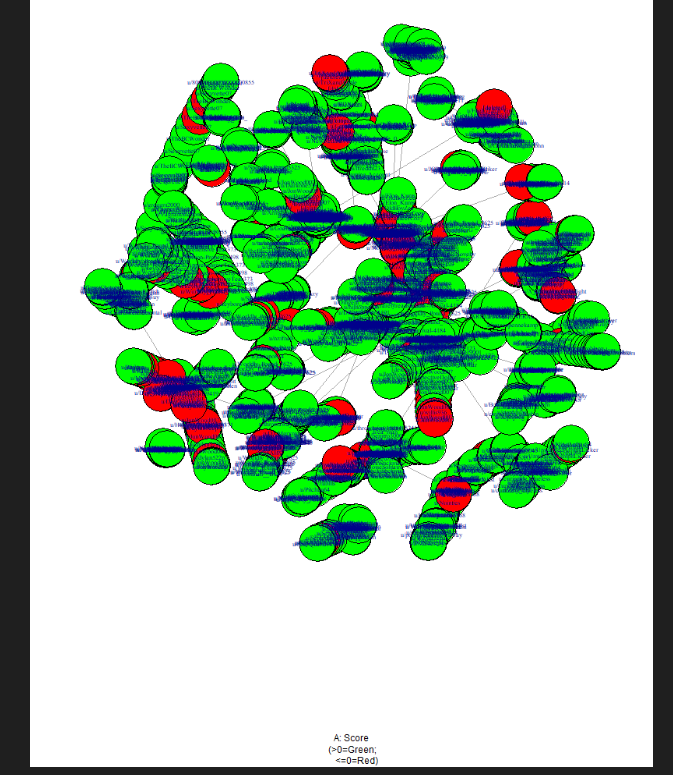
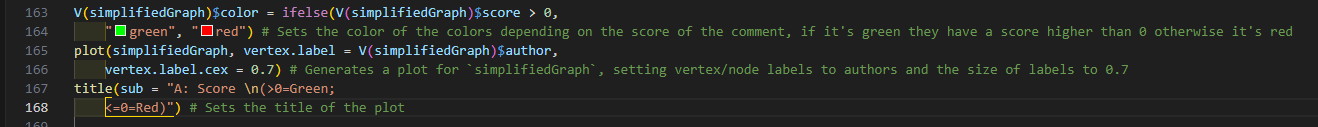
\*R would not output due to poor optimization/got hung up

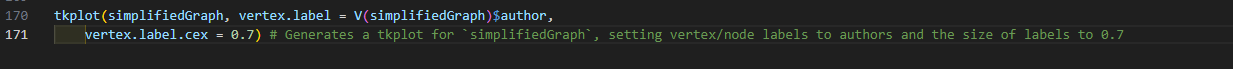


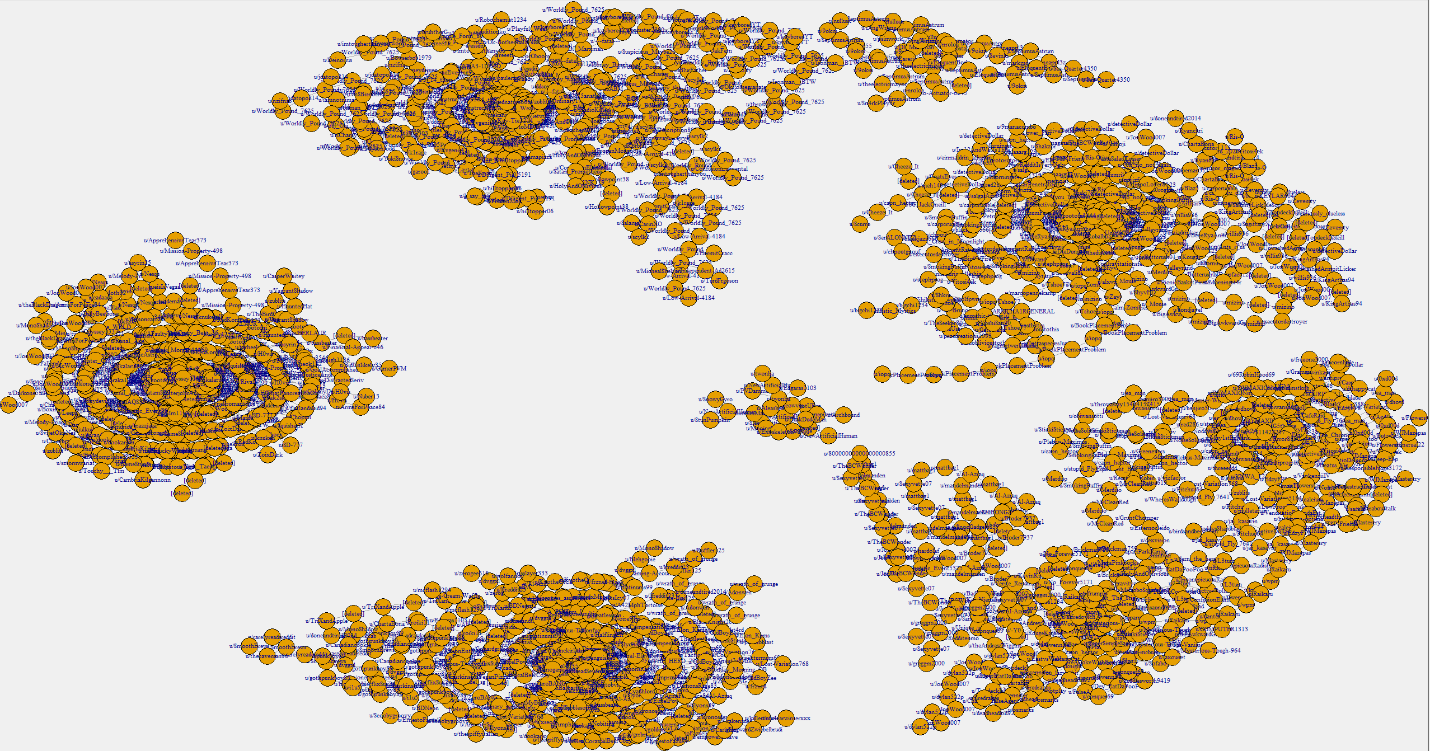


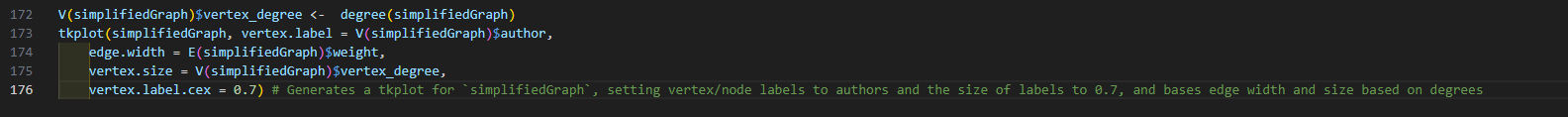


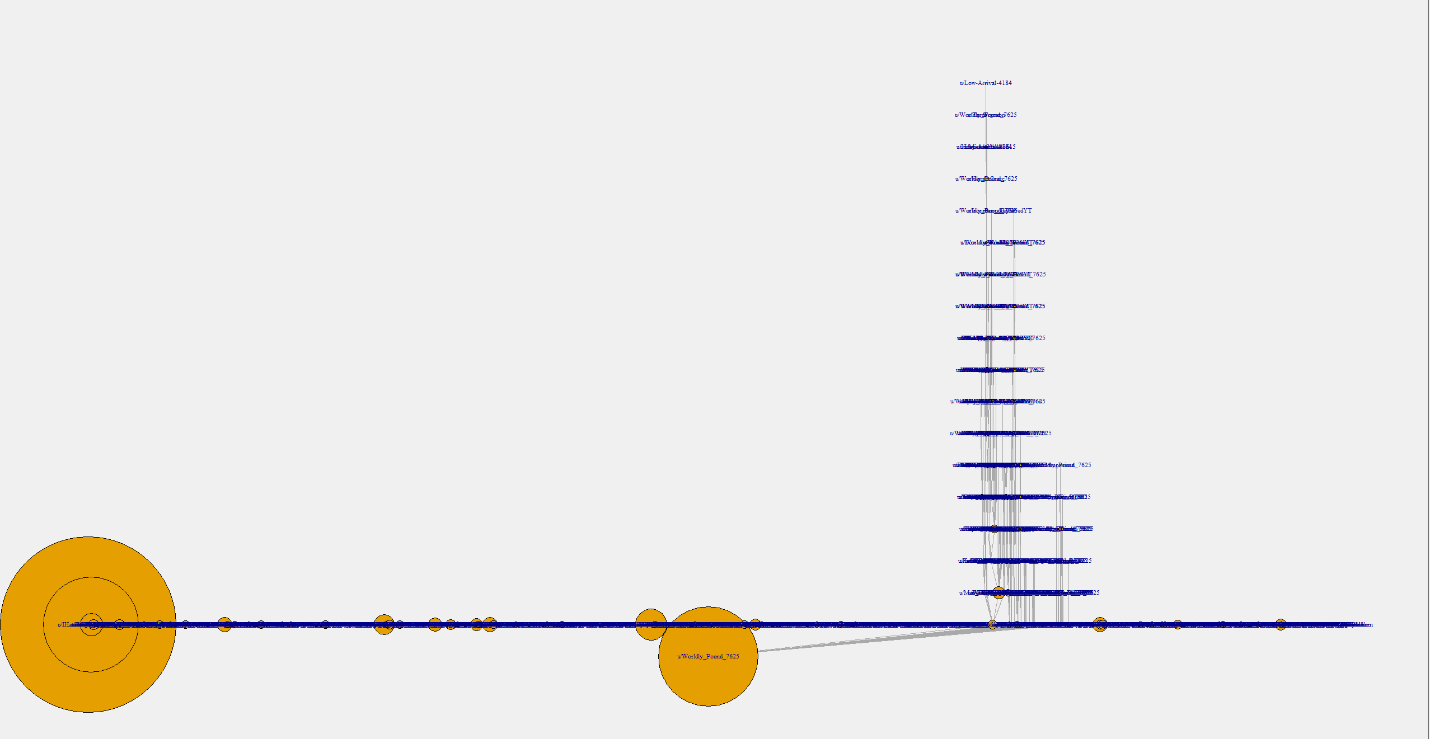
\*R refused to generate a plot due to poor optimization/got hung up

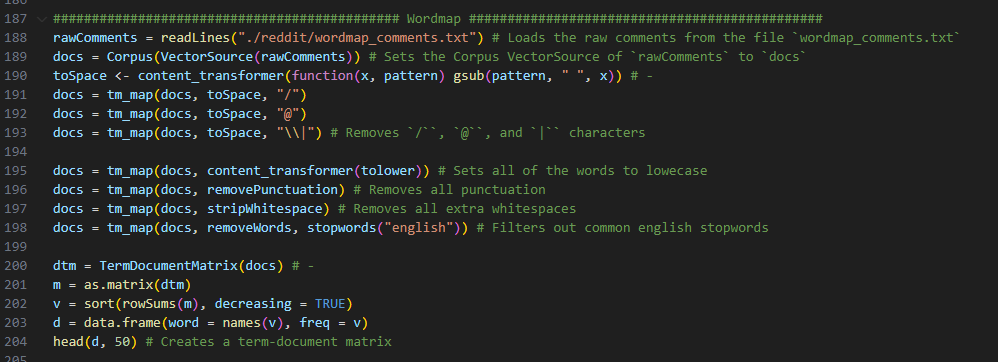




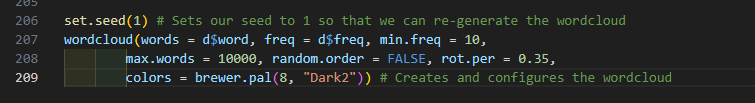














R seems to struggle working with my size of data for some reason, as a result of this, I tested to create some additional plots in Excel. Unfortunately, Excel ran into a few limitations for what I wanted to do. As a result of this, I performed manual analysis on the data on top of the analysis conducted in R to look for additional trends and concluded my data analysis.

**Step 6 – Results**

As this is supposed to serve as my technical documentation, I will keep my results limited here to a summary and my more detailed analysis of outcomes in my final paper. Looking at the results of data, analysis, and comments I am able to determine a few trends:

Intel – Most people so far seem to be happy with Intel graphics cards at the Arc series, specifically because of the price to performance ratio. The reason people are still hesitant to buy Intel dedicated graphics cards is because of the poor drivers and how early they are in development. Most people say that they would buy one in a year or two.

Nvidia – A large majority of people are disappointed in Nvidia’s price to performance ratios. Over the past 10 years the prices have climbed exponentially, and the performance has climbed but not enough to make up for the climb in price. Most people do agree, however, that their high-end card (currently the 4090) is the best performance graphics card but do also agree it may not be worth the price unless you only need it.

Amd – Most people agree that Amd currently has the best budget to mid-range GPUs that perform well enough compared to Nvidia.

Source code:

###############################################

# Dalton Murray #

# 06/18/2023 #

# Introduction to Social Media Data Analytics #

# Final Research Project #

###############################################

############################################# Note ##############################################

# Everything in this document is uncommented and has code to work with my environment.

# In order to successfully run it you may need to comment out lines of code, such as having multiple plots may not work in your environment all at once.

# You will also need to change my exact paths for `primaryComments` and `primaryPosts` to yours.

############################################# Setup ##############################################

# We need the library igraph

# install.packages("igraph") # Install `igraph` if not already installed

# install.packages("ggplot2") # Install `ggplot2` if not already installed

# install.packages("tm") # Install `tm` if not already installed

# install.packages("SnowballC") # Install `SnowballC` if not already installed

# install.packages("wordcloud") # Install `wordcloud` if not already installed

# install.packages("RColorBrewer") # Install `rColorBrewer` if not already installed

library("igraph") # Loads library `igraph`

library("ggplot2") # Loads library `ggplot2`

library("tm") # Loads library `tm`

library("SnowballC") # Loads library `SnowballC`

library("wordcloud") # Loads library `wordcloud`

library("RColorBrewer") # Loads library `RColorBrewer`

# Unfortunately I had to use exact paths instead of relative paths here because R was not liking using relative paths for this and would not find it if I used relative paths

primaryComments = read.csv("C:\\Users\\daltonPrimary\\Desktop\\LTU\\INT 6303 - Introduction to Social Media Data Analytics\\Final\\Project\\reddit\\Formatted\_comments\_combined.csv", header = T) # Importing `Formatted\_comments\_combined` data and telling it that it has headers, setting it to variable `primaryComments`

primaryPosts = read.csv("C:\\Users\\daltonPrimary\\Desktop\\LTU\\INT 6303 - Introduction to Social Media Data Analytics\\Final\\Project\\reddit\\Formatted\_primaryposts\_combined.csv", header = T) # Importing `Formatted\_primaryposts\_combined.csv` data and telling it that it has headers, setting it to variable `primaryPosts`

combinedGraph = graph\_from\_data\_frame(primaryComments, directed = F, vertices = primaryPosts) # Setting variable `combinedGraph` to the data from the data frame from `primaryComments` while telling it to be not directed, and we set its vertices to be equal to `primaryPosts`

simplifiedGraph = simplify(combinedGraph) # Sets `simplifiedGraph` to simplify `combinedGraph` removing self-ties/loops or duplicate edges

############################################ Standard functions ##############################################

LegOrder = data.frame(cbind(V(simplifiedGraph)$author)) # Creates the leg order of the `simplifiedGraph` and sets it to use `author`

write.csv(LegOrder, file = "./legOrder.csv") # Writes to file `legOrder.csv` the leg order

verticesSimplified = V(simplifiedGraph) # Sets the variable `verticesSimplified` to the vertices of `simplifiedGraph`

cat("Vertices of simplifiedGraph: \n") # Outputs

print(verticesSimplified) # Outputs variable

edgesSimplified = E(simplifiedGraph) # Sets the variable `edgesSimplified` to the vertices of `simplifiedGraph`

cat("Edges of simplifiedGraph: \n") # Outputs

print(edgesSimplified) # Outputs variable

gDensitysimplifiedGraph = graph.density(simplifiedGraph) # Sets the graph density of `simplifiedGraph` to `gDensitysimplifiedGraph`

cat("Graph Density simplifiedGraph:", gDensitysimplifiedGraph, "\n") # Prints out for organization

diametersimplifiedGraph = diameter(simplifiedGraph) # Sets the diameter/maximum geodesic distance of `simplifiedGraph` to `diametersimplifiedGraph`

cat("Diameter simplifiedGraph: ", diametersimplifiedGraph, "\n") # Prints out for organization

farthestNodesimplifiedGraph = farthest.nodes(simplifiedGraph) # Sets the farthest nodes/the diameter by nodes DE and PG and tells us geodesic distance between the nodes of `simplifiedGraph` to `farthestNodesimplifiedGraph`

cat("Farthest node simplifiedGraph: \n") # Prints out for organization

print(farthestNodesimplifiedGraph) # Prints out the information in `farthestNodesimplifiedGraph`

reciprocitysimplifiedGraph = reciprocity(simplifiedGraph) # Sets the reciprocity of `simplifiedGraph` to `recriprocitysimplifiedGraph`

cat("Recriprocity simplifiedGraph: ", reciprocitysimplifiedGraph, "\n") # Prints out for organization

transitivitysimplifiedGraph = transitivity(simplifiedGraph) # Sets the transitivity of `simplifiedGraph` to `transitivitysimplifiedGraph`

cat("Transitivity simplifiedGraph: ", transitivitysimplifiedGraph, "\n") # Prints out for organization

degreesimplifiedGraph = degree(simplifiedGraph) # Sets the degree of `simplifiedGraph` to `degreesimplifiedGraph`

cat("Degree simplifiedGraph: \n") # Prints out for organization

print(degreesimplifiedGraph) # Prints data in `degreesimplifiedGraph`

degreeInsimplifiedGraph = degree(simplifiedGraph, mode = "in") # Sets the degree of `simplifiedGraph` with mode `in` to `degreeInsimplifiedGraph`

cat("Degree In simplifiedGraph: \n") # Prints out for organization

print(degreeInsimplifiedGraph) # Prints data in `degreeInsimplifiedGraph`

degreeOutsimplifiedGraph = degree(simplifiedGraph, mode = "out") # Sets the degree of `simplifiedGraph` with mode `out` to `degreeOutsimplifiedGraph`

cat("Degree Out simplifiedGraph: \n") # Prints out for organization

print(degreeOutsimplifiedGraph) # Prints data in `degreeOutsimplifiedGraph`

evcentsimplifiedGraph = evcent(simplifiedGraph) # Sets the evcent of `simplifiedGraph` to `evcentsimplifiedGraph`

cat("Evcent simplifiedGraph: \n") # Prints out for organization

print(evcentsimplifiedGraph) # Prints data in `evcentsimplifiedGraph`

evcentVectorsimplifiedGraph = evcent(simplifiedGraph)$vector # Sets the evcent to be filtered for `vector` and sets it to `evcentVectorsimplifiedGraph`

cat("Evcent Vector simplifiedGraph: \n") # Prints out for organization

print(evcentVectorsimplifiedGraph) # Prints data in `evcentVectorsimplifiedGraph`

constraintsimplifiedGraph = constraint(simplifiedGraph) # Sets the constraints of `simplifiedGraph` to `constraintsimplifiedGraph`

cat("Constraint simplifiedGraph: \n") # Prints out for organization

print(constraintsimplifiedGraph) # Prints the data in `constraintsimplifiedGraph`

shortestPathssimplifiedGraph = shortest.paths(simplifiedGraph) # Sets the shortest paths of `simplifiedGraph` to `shortestPathssimplifiedGraph`

cat("Shortest paths simplifiedGraph: \n") # Prints out for organization

print(shortestPathssimplifiedGraph) # Prints the data in `shortestPathssimplifiedGraph`

# Analyzing ego metrics better, collating different metrics and binding columns to vector outputs

simplifiedGraphMetrics = data.frame(cbind(degree(simplifiedGraph))) # Gets degree

simplifiedGraphMetrics = cbind(simplifiedGraphMetrics, data.frame(cbind(betweenness(simplifiedGraph)))) # Gets betweenness and collates

simplifiedGraphMetrics = cbind(simplifiedGraphMetrics, data.frame(cbind(evcent(simplifiedGraph)$vector))) # Gets evcent and collates

simplifiedGraphMetrics = cbind(simplifiedGraphMetrics, data.frame(cbind(constraint(simplifiedGraph)))) # Gets constraints and collates

colnames(simplifiedGraphMetrics) = c("Degree", "Betweenness", "Eigenvecto", "Constraint") # Sets the columnn names

write.csv(simplifiedGraphMetrics, "./EgoMetrics.csv") # Writes the collated data to `EgoMetrics.csv`

subcomponent1simplifiedGraph = subcomponent(simplifiedGraph, 1) # Sets the reachability of id `1` in `simplifiedGraph` to `subComponent1simplifiedGraph`

cat("Subcomponent 1 simplifiedGraph: \n") # Prints for organization

print(subcomponent1simplifiedGraph) # Prints the data in `subComponent1simplifiedGraph`

subcomponent8simplifiedGraph = subcomponent(simplifiedGraph, 8) # Sets the reachability of id `8` in `simplifiedGraph` to `subComponent1simplifiedGraph`

cat("Subcomponent 8 simplifiedGraph: \n") # Prints for organization

print(subcomponent8simplifiedGraph) # Prints the data in `subComponent8simplifiedGraph`

cliquessimplifiedGraph = cliques(simplifiedGraph, min = 2, max = 5) # Sets variable `cliquessimplifiedGraph` to the cliques of `simplifiedGraph` with min and max of 5

write.csv(cliquessimplifiedGraph, "./cliquesSimplified.csv") # Writes the collated data to `cliquesSimplified.csv`

cliquesNumsimplifiedGraph = clique.number(simplifiedGraph) # Sets variable `cliquesNumsimplifiedGraph` to the largest clique in the graph

cat("Cliques Number:", cliquesNumsimplifiedGraph, "\n") # Prints out with formatting

largestCliquessimplifiedGraph = largest.cliques(simplifiedGraph) # Sets variable `largestCliquessimplifiedGraph` to the largest collection of connected vertices

cat("Largest Cliques: \n") # Prints out for organization

print(largestCliquessimplifiedGraph) # Prints out variable `largestCliquessimplifiedGraph`

assortativitysimplifiedGraph = assortativity(simplifiedGraph, primaryPosts$score, directed = F) # Sets variable `assortativitysimplifiedGraph` to the value of the assortativity coefficient, calculating whether tenure in Congress is a basis for homophily

cat("Assortativity:", assortativitysimplifiedGraph, "\n") # Prints out variable `assortativitysimplifiedGraph`

assortativityDegreesimplifiedGraph = assortativity\_degree(simplifiedGraph, directed = F) # Sets variable `assortativityDegreesimplifiedGraph` to the value of the assortativity degree

cat("Assortativity Degree:", assortativityDegreesimplifiedGraph, "\n")

clusterWalkTrapsimplifiedGraph = cluster\_walktrap(simplifiedGraph) # Sets variable `clusterWalkTrapsimplifiedGraph` to the value of the cluster walk trap

cat("Cluster Walktrap: \n")

print(clusterWalkTrapsimplifiedGraph)

clusterWalkTrap4simplifiedGraph = cluster\_walktrap(simplifiedGraph) # Sets variable `clusterWalkTrap4simplifiedGraph` to the value of the cluster walk trap

cat("Cluster Walktrap 4: \n")

print(clusterWalkTrap4simplifiedGraph[4]) # Outptus specifically the 4th group of walktrap

clusterWalkTrapStepssimplifiedGraph = cluster\_walktrap(simplifiedGraph, steps = 200) # Sets variable `clusterWalkTrapStepssimplifiedGraph` to the value of the cluster walk trap with 200 steps

cat("Cluster Walktrap Steps: \n")

print(clusterWalkTrapStepssimplifiedGraph)

clusterOptimalsimplifiedGraph = cluster\_optimal(simplifiedGraph) # Sets variable `clusterOptimalsimplifiedGraph` to the cluster oiptimal of `simplifiedGraph`

partitionssimplifiedGraph = data.frame(cbind(clusterOptimalsimplifiedGraph$author, clusterOptimalsimplifiedGraph$score)) # Sets variable `partitionsimplifiedGraph` to the data frame of cluster optimal author and score

columnsimplifiedGraph = colnames(partitionssimplifiedGraph) = c("Comment", "Group") # Sets `columnsimplifiedGraph` to the column names specified

cat("Partitions: \n")

print(partitionssimplifiedGraph) # Outputs

print(columnsimplifiedGraph) # Outputs

############################################# Plots & Graphs ##############################################

set.seed(1) # Setting a seed to the value of 1 so that everytime I generate a plot or graph it stays the same rather than randomly generating a new one each time

plot(simplifiedGraph, vertex.label = V(simplifiedGraph)$author, vertex.label.cex = 0.7) # Generates a plot for `simplifiedGraph`, setting vertex/node labels to authors and the size of labels to 0.7

plot(simplifiedGraph, layout = layout\_as\_tree(simplifiedGraph),

vertex.size = evcent(simplifiedGraph)$vector \* 5^1.5,

vertex.label = V(simplifiedGraph)$author) # Plots `simplifiedGraph` with a tree layout and sets vertex/nodes to label author with vector resizing

plot(simplifiedGraph, layout = layout\_with\_gem(simplifiedGraph),

vertex.size = evcent(simplifiedGraph)$vector \* 5^1.5,

vertex.label = V(simplifiedGraph)$author) # Plots `simplifiedGraph` with a gem layout and sets vertex/nodes to label author with vector resizing

walkTrapPlot = plot(clusterWalkTrapStepssimplifiedGraph, simplifiedGraph) # Sets variable `walkTrapPlot` to the plot of the variable `clusterWalkTrapStepssimplifiedGraph` with `simplifiedGraph`

cat("Walk trap plot:")

print(walkTrapPlot) # Prints out the plot

tkplot(clusterOptimalsimplifiedGraph, simplifiedGraph, margin = -0.05,

edge.color = "darkred", vertex.label = V(simplifiedGraph)$primaryPosts,

vertex.size = evcent(simplifiedGraph)$vector \* 5,

vertex.shape = "sphere",

vertex.label.family = "sans", vertex.label.font = 2,

vertex.label.cex = 0.7, vertex.label.dist = 0.25) # Creates and configures a plot

V(simplifiedGraph)$color = ifelse(V(simplifiedGraph)$score > 0,

"green", "red") # Sets the color of the colors depending on the score of the comment, if it's green they have a score higher than 0 otherwise it's red

plot(simplifiedGraph, vertex.label = V(simplifiedGraph)$author,

vertex.label.cex = 0.7) # Generates a plot for `simplifiedGraph`, setting vertex/node labels to authors and the size of labels to 0.7

title(sub = "A: Score \n(>0=Green;

<=0=Red)") # Sets the title of the plot

tkplot(simplifiedGraph, vertex.label = V(simplifiedGraph)$author, vertex.label.cex = 0.7) # Generates a tkplot for `simplifiedGraph`, setting vertex/node labels to authors and the size of labels to 0.7

V(simplifiedGraph)$vertex\_degree <- degree(simplifiedGraph)

tkplot(simplifiedGraph, vertex.label = V(simplifiedGraph)$author,

edge.width = E(simplifiedGraph)$weight,

vertex.size = V(simplifiedGraph)$vertex\_degree,

vertex.label.cex = 0.7) # Generates a tkplot for `simplifiedGraph`, setting vertex/node labels to authors and the size of labels to 0.7, and bases edge width and size based on degrees

############################################# Wordmap ##############################################

rawComments = readLines("./reddit/wordmap\_comments.txt") # Loads the raw comments from the file `wordmap\_comments.txt`

docs = Corpus(VectorSource(rawComments)) # Sets the Corpus VectorSource of `rawComments` to `docs`

toSpace <- content\_transformer(function(x, pattern) gsub(pattern, " ", x)) # -

docs = tm\_map(docs, toSpace, "/")

docs = tm\_map(docs, toSpace, "@")

docs = tm\_map(docs, toSpace, "\\|") # Removes `/``, `@``, and `|`` characters

docs = tm\_map(docs, content\_transformer(tolower)) # Sets all of the words to lowecase

docs = tm\_map(docs, removePunctuation) # Removes all punctuation

docs = tm\_map(docs, stripWhitespace) # Removes all extra whitespaces

docs = tm\_map(docs, removeWords, stopwords("english")) # Filters out common english stopwords

dtm = TermDocumentMatrix(docs) # -

m = as.matrix(dtm)

v = sort(rowSums(m), decreasing = TRUE)

d = data.frame(word = names(v), freq = v)

head(d, 50) # Creates a term-document matrix

set.seed(1) # Sets our seed to 1 so that we can re-generate the wordcloud

wordcloud(words = d$word, freq = d$freq, min.freq = 10,

max.words = 10000, random.order = FALSE, rot.per = 0.35,

colors = brewer.pal(8, "Dark2")) # Creates and configures the wordcloud

**References**

Anyan, F. (2013). The Influence of Power Shifts in Data Collection and Analysis Stages: A Focus on Qualitative Research Interview. <https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2624458/Anyan.pdf?sequence=2>

Lai, J. (n.d.). Josephlai241/urs: Universal reddit scraper - a comprehensive reddit scraping/archival command-line tool. GitHub. <https://github.com/JosephLai241/URS>

Medvedev, A., Lambiotte, R., & Delvenne, J.-C. (2020, June). The anatomy of Reddit: An overview of academic research. <https://arxiv.org/pdf/1810.10881.pdf>

Proferes, N., Jones, N., Gilbert, S., Fiesler, C., & Zimmer, M. (2021). Studying Reddit: A Systematic Overview of Disciplines, Approaches, Methods, and Ethic. <https://journals.sagepub.com/doi/pdf/10.1177/20563051211019004>

Reddit - GPU Pricing Update: Hardware Still Overpriced?. Reddit. (n.d.-a). <https://www.reddit.com/r/hardware/comments/10jd9xf/gpu_pricing_update_hardware_still_overpriced/>

Reddit - GPUs are very expensive nowadays, what would be a reasonable price for a low, mid and high end GPU for you?. Reddit. (n.d.-a). <https://www.reddit.com/r/gaming/comments/12vl57c/gpus_are_very_expensive_nowadays_what_would_be_a/>

Reddit - Most PC hardware pricing is back to normal so why not GPUs?. Reddit. (n.d.-c). <https://www.reddit.com/r/pcgaming/comments/128r3j5/most_pc_hardware_pricing_is_back_to_normal_so_why/>

Reddit - price of GPU are insane. Reddit. (n.d.). <https://www.reddit.com/r/pcgaming/comments/105lgwm/price_of_gpu_are_insane/>

Reddit - What are your thoughts on value for money on current gen cards (RTX 40/RX 7000) for RT gaming? Which is better?. Reddit. (n.d.-f). <https://www.reddit.com/r/buildapc/comments/124tz2b/what_are_your_thoughts_on_value_for_money_on/>

Reddit - What’s going on with Nvidia GPU prices? Reddit. (n.d.-c). <https://www.reddit.com/r/buildapc/comments/126rjun/whats_going_on_with_nvidia_gpu_prices/>

Reddit - Why are graphic card prices still so high if they are not selling?. Reddit. (n.d.-e). <https://www.reddit.com/r/hardware/comments/1134ddq/why_are_graphic_card_prices_still_so_high_if_they/>

Reddit - [HUB] Nvidia and AMD Keep Dropping GPU Prices - June GPU Pricing Update. Reddit. (n.d.-a). <https://www.reddit.com/r/hardware/comments/14ec8zx/hub_nvidia_and_amd_keep_dropping_gpu_prices_june/>

I pledge that on all academic work that I submit, I will neither give nor receive unauthorized aid, nor will I present another person's work as my own.

Dalton Murray