Taller 1: Diferenciar métodos para recolectar información de redes

Apellidos	Nombre	Código	Login
Arenas Solano	Karol Daniela	202014996	k.arenas
Fonseca Najar	Haider Yesid	202116856	h.fonseca
Urrea Lopez	Juan Sebastian	201914710	js.urrea

1. Tamaño de la muestra:

Para determinar el tamaño de la muestra se procede en primera instancia a calcular las proporciones de sexo femenino en la población. Esto da como resultado una proporción de mujeres de 0.4940 (p) y de hombres igual a 1 – p. Seguidamente, utilizamos una combinación de las fórmulas de Cochran y de Slovin dado que sabemos la proporción exacta de la población y tenemos un valor deseado del margen de error y nivel de confianza. Se obtiene que el tamaño de la muestra es de **239**.

```
# Proporción de mujeres
m = nodos.groupby("Gender").count()["Age"]
N = m["Male"]+m["Female"]      # Poblacion Total
p = m["Female"]/N              # Proporcion girls
p
```

```
0.49407863571766936
```

Imagen 1 . Cálculo de la proporción de mujeres.

```
#Tamaño de la muestra
q = 1-p
e = 0.05
alpha = 0.05
Z = norm.ppf(alpha)
n = ((Z**2)*p*q)/(e**2)
n = N*n/(N+n-1)
n = math.floor(n)
n
```

```
239
```

Imagen 2. Cálculo del tamaño de la muestra.

2. Muestreo estratificado de acuerdo con el sexo:

Se realiza un muestreo estratificado de acuerdo al sexo en donde se determina el promedio de edad de cada género.

```
# Promedio edad muestra mujeres
n_mujeres = int(n*p)
muestra_mujeres = nodos[nodos["Gender"]=="Female"].sample(n_mujeres)
muestra_mujeres["Age"].mean()
```

```
23.37084869491525
```

```
# Promedio edad muestra hombres
n_hombres = n - n_mujeres
muestra_hombres = nodos[nodos["Gender"]=="Male"].sample(n_hombres)
muestra_hombres["Age"].mean()
```

```
24.458531173553716
```

Imagen 3. Promedios de edad por género

3. Construcción de la red por muestreo estadístico estratificado:

Se utiliza NetworkX para construir un grafo dirigido con base en los nodos y arcos calculados en las variables ‘muestra’ y ‘arcos_muestra’. Los nodos se extraen de los índices de ‘muestra’, y los arcos se crean conectando los nodos de origen y destino extraídos de cada fila de ‘arcos_muestra’. Los nodos son tanto la muestra de mujeres como la de hombres y los arcos son todos aquellos cuyo origen es un nodo de los mencionados anteriormente.

```
## Construir Red
G1 = nx.DiGraph()
G1.add_nodes_from(muestra.index.tolist())

for index, row in arcos_muestra.iterrows():
    source = row['Source']
    target = row['Target']
    G1.add_edge(source, target)
```

Imagen 4. Construcción de la red (estratificado)

```
# Top 5 degree
dic_grados= G1.degree()
sorted_deg = sorted(dic_grados,key=lambda item:item[1],reverse=True)
top_5_G1 = [item[0] for item in sorted_deg[:5]]
muestra.loc[top_5_G1]
```

id	Gender	Age	Height	Weight
354	Female	17.000000	1.600000	59.000000
39	Female	21.000000	1.750000	88.000000
1486	Male	17.412629	1.723191	97.350366
1316	Male	39.656559	1.789992	98.021766
817	Male	22.740275	1.717288	75.948164

Imagen 5. Top 5 de niños con más conexiones

4. Muestreo por método de la bola de nieve:

Primero se obtuvo una nueva columna que contiene la lista de las conexiones de cada nodo. Luego, se realiza el muestreo por bola de nieve partiendo de dos nodos aleatorios. Finalmente, se obtiene el promedio de edades por género de la muestra.

```
# Crear un diccionario para almacenar las conexiones de cada nodo
nodo_conexiones = {}

# Llenar el diccionario con las conexiones
for index, row in arcos.iterrows():
    source = row['Source']
    target = row['Target']

    if source in nodo_conexiones:
        nodo_conexiones[source].append(target)
    else:
        nodo_conexiones[source] = [target]

# Crear una lista de conexiones para cada nodo
nodos['conexiones'] = nodos.index.to_series().apply(lambda x: nodo_conexiones.get(x, []))
nodos
```

Imagen 6. Creación lista de conexiones

```
def snowball():
    ultima_ola = nodos.sample(2).index.to_list()
    nodos_snowball = {i for i in ultima_ola}
    while True:
        conexiones = nodos.loc[ultima_ola, "conexiones"]
        ultima_ola = set()
        for source, lista_arcos in conexiones.iteritems():
            for target in lista_arcos:
                if target not in nodos_snowball: # Agrego un nuevo nodo
                    if len(nodos_snowball) == n: # No puedo añadir más
                        return nodos_snowball
                    nodos_snowball.add(target)
                    ultima_ola.add(target)

nodos_snowball = snowball()
arcos_snowball = arcos[arcos["Source"].isin(nodos_snowball)]
nodos_snowball = nodos[nodos.index.isin(nodos_snowball)]
```

Imagen 7. Muestreo por bola de nieve

```
In [33]: #Edad promedio niñas
         nodos_snowball[nodos_snowball["Gender"]=="Female"]["Age"].mean()
Out[33]: 24.72509500000001

In [34]: #Edad promedio niños
         nodos_snowball[nodos_snowball["Gender"]=="Male"]["Age"].mean()
Out[34]: 24.038920704697993
```

Imagen 8. Promedio de edades

5. Construcción de la red por muestreo de bola de nieve:

De igual forma al muestreo estadístico estratificado, se crea la red a partir de los nodos muestreados y todos los arcos cuyo origen es alguno de dichos nodos. Luego, se calcula el Top 5 de niños con mayor grado en la muestra.

```
## Construir Red
G2 = nx.DiGraph()
G2.add_nodes_from(nodos_snowball.index.tolist())

for index, row in arcos_snowball.iterrows():
    source = row['Source']
    target = row['Target']
    G2.add_edge(source, target)
```

Imagen 9. Construcción de la red (snowball)

```
# Top 5 degree
dic_grados= G2.degree()
sorted_deg = sorted(dic_grados,key=lambda item:item[1],reverse=True)
top_5_G2 = [item[0] for item in sorted_deg[:5]]
nodos_snowball.loc[top_5_G2]
```

id	Gender	Age	Height	Weight	conexiones
272	Male	18.000000	1.700000	60.000000	[1045, 1669, 922, 503, 1933, 194, 1575, 935, 7...
701	Male	17.282945	1.821514	59.605028	[35, 28, 1781, 1538, 1350, 1837, 495, 1699, 15...
895	Male	21.037514	1.636592	70.000000	[2015, 1394, 1074, 1159, 961, 502, 1648, 376, ...
1334	Female	22.884722	1.622787	82.000000	[793, 260, 1675, 1679, 1243, 1641, 746, 194, 9...
419	Male	29.000000	1.620000	89.000000	[2042, 1212, 584, 426, 73, 103, 2060, 1313, 11...

Imagen 10. Top 5 de niños con más conexiones

6. Selección de método de muestreo:

Se deben elegir el grafo por muestreo en redes. La elección del método de muestreo en redes es crucial, y en este contexto, se destaca tras analizar diversas métricas como Degree, Centrality, Betweenness, Closeness, Page Rank y Clustering. Notablemente, los cinco mejores nodos identificados mediante el muestreo en redes presentan puntuaciones superiores a los cinco nodos seleccionados mediante un enfoque estadístico de muestreo. Estas observaciones resaltan la eficacia del muestreo en redes para capturar de manera más efectiva nodos significativos y relevantes en comparación con el muestreo estadístico tradicional. El archivo exportado es *graph.gexf*

Métrica	Muestreo	Top 1	Top 2	Top 3	Top 4	Top 5
Degree	Estadístico	22	22	22	22	22
	Red	25	23	23	23	22
Centrality	Estadístico	0.0	0.0	0.0	0.0	0.052554
	Red	0.071535	0.060897	0.037122	0.040932	0.039618
Betweenness	Estadístico	0.00012	0.000317	0.000119	0.000213	0.00018
	Red	0.00098	0.010383	0.014827	0.004951	0.003573
Closeness	Estadístico	0.002997	0.002923	0.002302	0.003401	0.003728
	Red	0.013527	0.013117	0.009635	0.011358	0.011948
Page Rank	Estadístico	0.000803	0.000656	0.000636	0.000631	0.000622
	Red	0.000826	0.000838	0.000758	0.000798	0.000753
Clustering	Estadístico	0	0	0	0	0
	Red	0	0	0.001976	0.001976	0

Tabla 1. Medidas de centralidad por tipo de muestreo

7. Exploración del grafo en Gephi

La red es dirigida debido a la presencia de relaciones unidireccionales en los datos. Al analizar los datos, se observa que algunos niños tienen arcos de amistad con otros, pero estas amistades no son necesariamente recíprocas. Esta asimetría en las conexiones sugiere que hay una dirección definida en las relaciones. Esta dinámica puede indicar jerarquías en las amistades, influencia social diferencial y complejas interacciones entre los niños. Además, la naturaleza de los arcos es indicar con quién suele pasar más tiempo un niño, lo cual no es una relación simétrica ya que lo que un niño puede considerar como mucho tiempo puede diferir de lo que piensan otros.

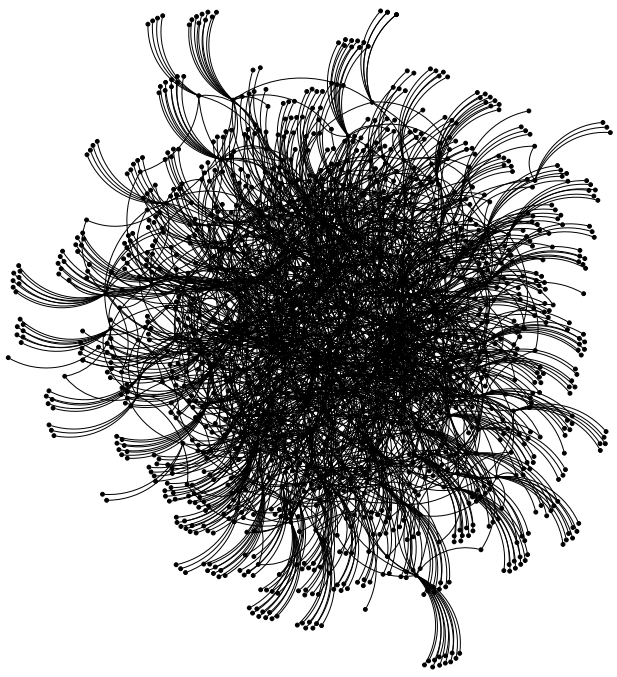

Imagen 11. Visualización de la red con el algoritmo Yifan Hu

El valor de modularidad de la red es de 0.594 y el número de comunidades identificadas es 26. Dado que la modularidad tiene un valor cercano a 1, esto indica que la red tiene una estructura de agrupamiento relativamente fuerte, es decir que los nodos tienen más conexiones dentro de sus comunidades que entre ellas. Este es un resultado esperado, pues la metodología de recolección por bola de nieve puede crear comunidades distintas por cada nodo que es muestreado y que nomina sus conexiones.

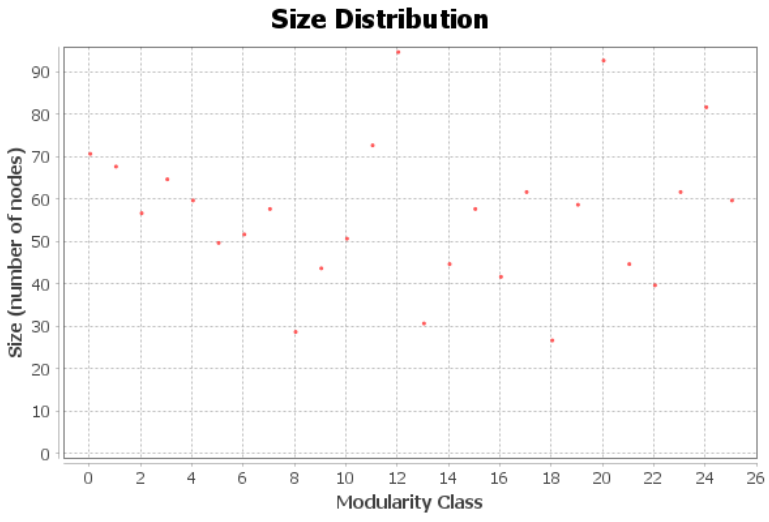

Imagen 12. Resultados del reporte de Modularidad

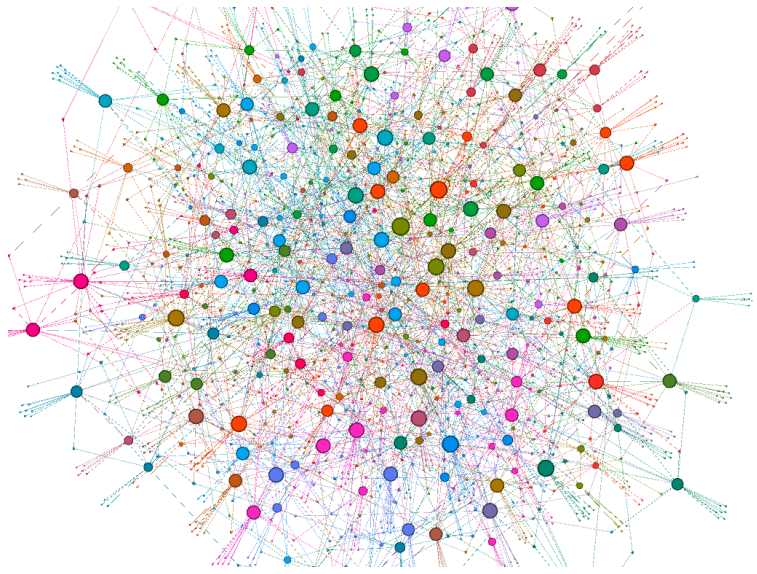

Imagen 13. Visualización de las comunidades identificadas.

Se realizó una partición de la red a partir de las comunidades identificadas. Sobre cada una, se calculó la suma de los grados de sus nodos y la densidad del subgrafo resultante. Para esto, se exportó de Gephi el archivo *nodos_gephi_final.csv* que contiene la comunidad identificada para cada nodo. Luego, se calcularon sus métricas por comunidad utilizando Excel y Python.

En cuanto a la suma de los grados, esta métrica mide el número de amistades de los nodos de la comunidad tanto dentro como fuera de esta. Para la muestra utilizada, se obtuvo que las comunidades con mayor valor de esta métrica fueron la número 12 y la número 20, con un total de 370 y 342 respectivamente. Dichas comunidades también son las más grandes identificadas, sobrepasando la cantidad de 90 nodos por comunidad. En cuanto a las comunidades con menor valor de esta métrica, se encuentran la número 8 y la 18, ambas con un valor de 89.

Por su parte, los subgrafos se crearon eliminando todos los nodos (y arcos asociados) del grafo original que no eran parte de la comunidad a analizar. La densidad dio como resultado que las comunidades más conectadas son la número 8, 10 y 18, siendo el valor más alto de 0.0047; por su parte, las comunidades número 6, 9, 13, 15, 16, 22, 23 y 25 son las menos conectadas, todas con una densidad de 0. Teniendo en cuenta los resultados anteriores, se evidencia que es más fácil para una red pequeña tener una densidad alta comparado con una red grande.

```
mc = pd.read_csv("../nodos_gephi_final.csv")
finales = []
for i in range(26):
    mc_nodos = mc[mc.modularity_class==i]["Id"]
    G_mc = G.copy()
    list_nodos = list(G_mc.nodes)
    for nodo in list_nodos:
        if nodo not in mc_nodos:
            G_mc.remove_node(nodo)
    finales.append((i, nx.density(G_mc)))
```

Imagen 14. Cálculo de la densidad por comunidad