

Práctica 1

Ignacio Arias Barra

February 13, 2017

Análisis de grafos y Redes Sociales

Hoja de ejercicios 1: Teoría de grafos

El grafo out.ego-twitter almacena información acerca de usuarios de Twitter.

En concreto, se trata de un grafo dirigido no ponderado donde cada vértice representa a un usuario, mientras que una arista (v, u) indica que el usuario v sigue al usuario u . El objetivo de esta hoja de ejercicios es analizar la estructura de este grafo y extraer diferentes características estructurales del mismo.

1. Carga el grafo en memoria. Las funciones `read_table` y `graph.data.frame` facilitan la tarea. Buscar en la documentación de `igraph` y R su descripción y uso.

```
library(igraph)

## 
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
## 
##     decompose, spectrum

## The following object is masked from 'package:base':
## 
##     union

library(ggplot2)
library(ape)

## 
## Attaching package: 'ape'

## The following objects are masked from 'package:igraph':
## 
##     edges, mst, ring

library(sand)

## Loading required package: igraphdata

## 
## Statistical Analysis of Network Data with R
## Type in C2 (+ENTER) to start with Chapter 2.

sep = " "
file = './out.ego-twitter'

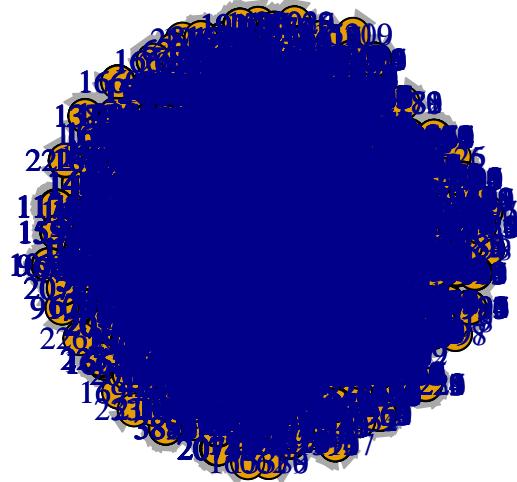
data <- read.table(file, sep = sep)
g <- graph.data.frame(data, directed=TRUE)
```

Con la función ‘`read.table`’ leemos el fichero `./out.ego-twitter` del actual directorio. Identificamos el separador como el que se emplea en el fichero para separar el dato vértice ‘ v ’ del dato vértice ‘ u ’. El resultado es un data frame.

La función ‘graph.data.frame’ transforma el dataframe obtenido en un grafo igraph. En este caso indicamos que es dirigido, ya que un usuario puede seguir a otro pero no ser recíproca esa relación.

2. Muestra el grafo por pantalla e indica qué tipo de grafo es. ¿Es de ayuda la visualización del grafo para identificar su tipo?

```
plot(g)
```



```
n_nodes <- length(V(g))
conn <- is.connected(g)
bip <- is.bipartite(g)
dir <- is.directed(g)
wei <- is.weighted(g)
nam <- is.named(g)
```

Se trata de un grafo de 23370 nodos, por lo que la representación general que se realizar con la función ‘plot(grafo)’ no ayudar para determinar con qué tipo de grafo estamos trabajando.

Al ejecutar las funciones ‘is.connected’, ‘is.bipartite’, ‘is.directed’, ‘is.weighted’ y ‘is.named’, obtenemos los siguientes valores respectivamente: *FALSE*, *FALSE*, *TRUE*, *FALSE*, *TRUE*. A partir de estos resultados obtenidos y teniendo en cuenta lo que simboliza el grafo, decimos que estamos ante un grafo no conexo, no bipartido, dirigido, no ponderado y cuyos vértices tienen un nombre simbólico asociado, en este caso números que identifican a usuarios de Twitter.

Que un grafo sea conexo indica que desde todos los nodos se puede llegar al resto. En este caso, al ser dirigido, tenemos que hay usuarios que no son alcanzables mediante las relaciones entre sus contactos debido a la no reciprocidad en la relación followers-following.

Es lógico pensar que un grafo con tantos nodos no sea bipartido, es decir, que no podamos separarlo en dos subgrafos obteniendo vértices que no tienen relación entre ellos.

Respecto a la ponderación, se está suponiendo que la relación que se mantiene entre dos usuarios es la misma, sin importar si un usuario es más importante que otro.

3. ¿Qué representa, en este grafo, el grado de un vértice? Analiza y comenta la distribución del grado de este grafo.

El grado de un vértice representa qué usuarios son seguidos o siguen a otros. Al ser un grafo dirigido, tenemos que diferenciar entre el caso en el que un vértice/usuario se une con otro en una dirección o en ambas. En el caso en el que un usuario esté conectado a otro, sin ser recíproca esa relación, es porque el primero es follower del segundo. En el caso en el que en la misma arista/relación los dos estén conectados, es porque ambos son followers entre sí.

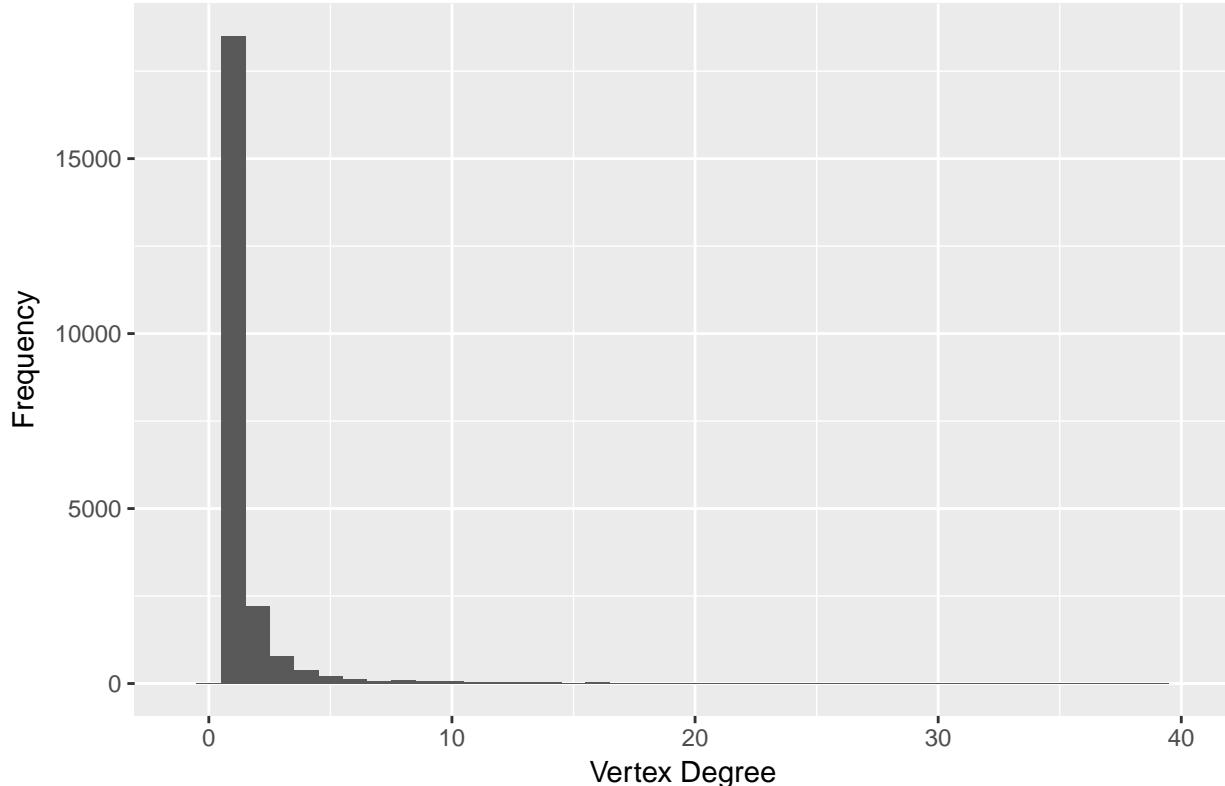
```

# Grado no dirigido
grado_g <- degree(g)
grado_gDF <- as.data.frame(grado_g)
ggplot(grado_gDF, aes(grado_gDF)) + geom_histogram(binwidth = 1) +
  xlab('Vertex Degree') + ylab('Frequency') + ggtitle('Histogram of total degree') +
  xlim(-1,40)

## Warning: Removed 306 rows containing non-finite values (stat_bin).

```

Histogram of total degree



El histograma anterior muestra el número de relaciones que hay entre los usuarios del grafo. Vemos que hay una gran cantidad de cuentas con pocos usuarios siguiendo o siendo seguidos por otros.

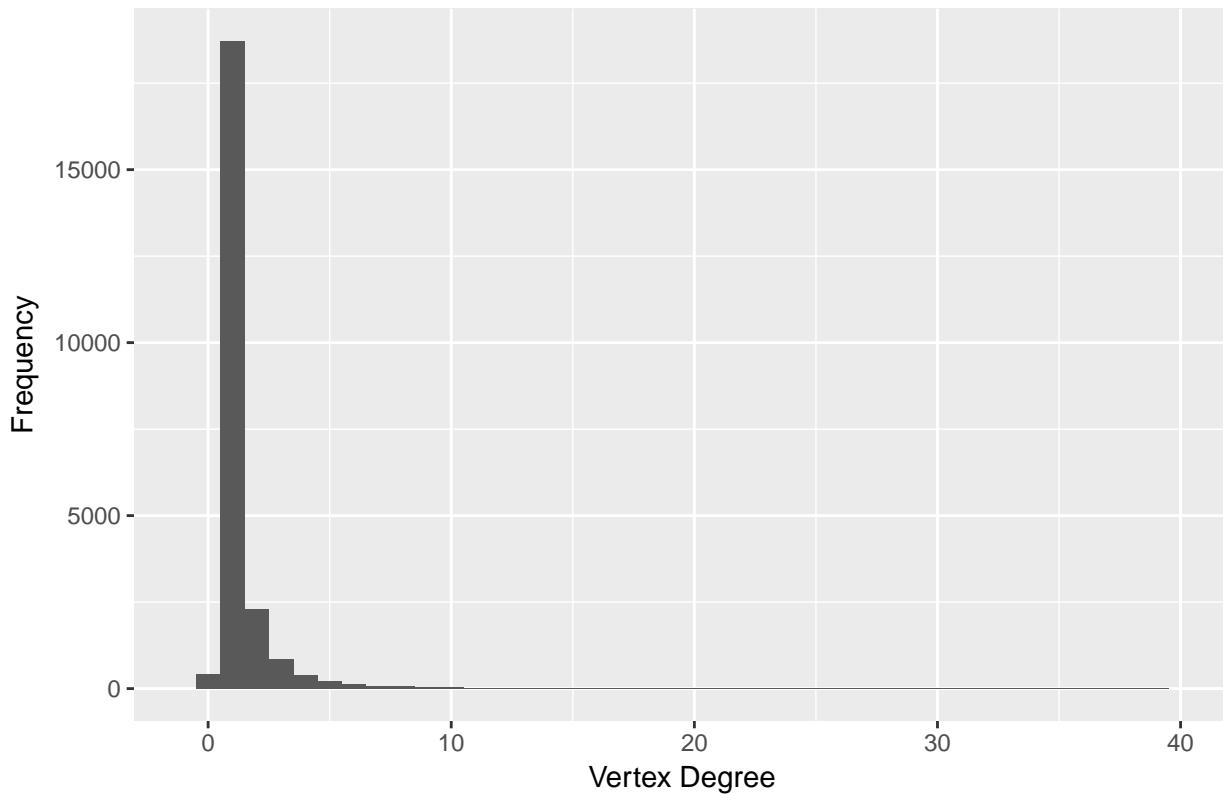
```

# Grado teniendo en cuenta followers de un usuario
grado_g_in <- degree(g, mode='in')
grado_gDF_in <- as.data.frame(grado_g_in)
ggplot(grado_gDF_in, aes(grado_gDF_in)) + geom_histogram(binwidth = 1) +
  xlab('Vertex Degree') + ylab('Frequency') + ggtitle('Histogram of input degree') +
  xlim(-1,40)

## Warning: Removed 4 rows containing non-finite values (stat_bin).

```

Histogram of input degree



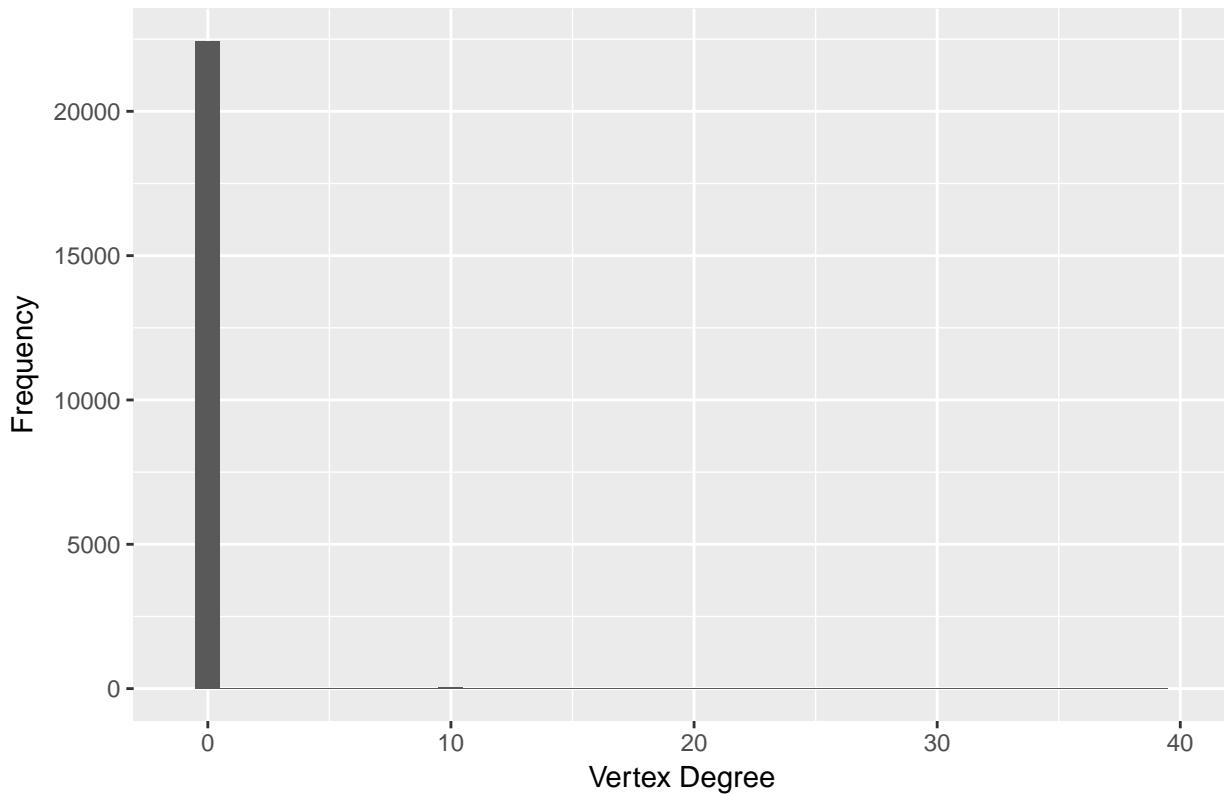
Este histograma muestra las conexiones entrantes en cada nodo, es decir, los followers que uno tiene en Twitter. Debido a que para un número alto de seguidores, la frecuencia es muy baja y puesto que así se visualiza mejor el histograma, limitamos el eje X a 40 followers.

Se puede apreciar que la tendencia general es a tener muchas cuentas con pocos followers.

```
# Grado teniendo en cuenta gente following por un usuario
grado_g_out <- degree(g, mode='out')
grado_gDF_out <- as.data.frame(grado_g_out)
ggplot(grado_gDF_out, aes(grado_gDF_out)) + geom_histogram(binwidth = 1) +
  xlab('Vertex Degree') + ylab('Frequency') + ggtitle('Histogram of output degree') +
  xlim(-1,40)

## Warning: Removed 292 rows containing non-finite values (stat_bin).
```

Histogram of output degree



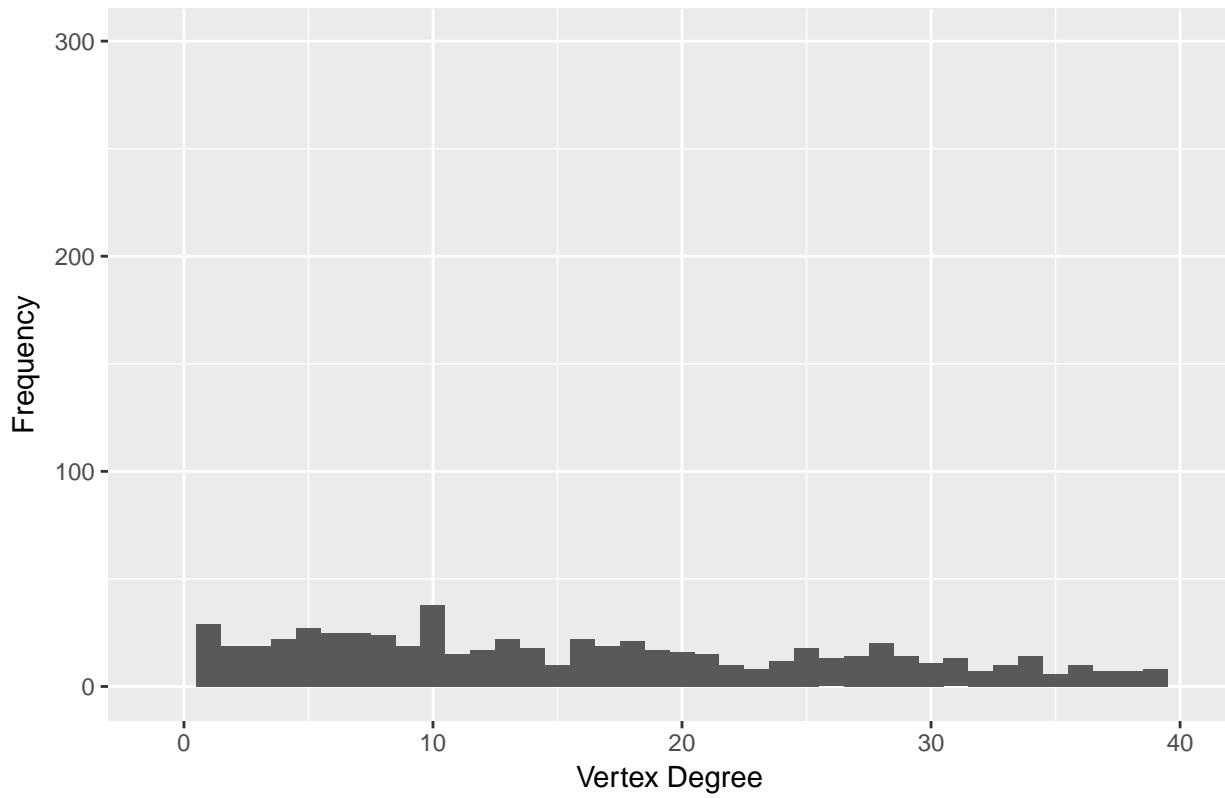
El histograma representado muestra la distribución de cuentas y a cuántas otras cuentas siguen en la plataforma. Se puede ver como hay muchas cuentas que no siguen a ninguna otra cuenta. Esto puede ser indicio de que se intente que en los análisis de grafos, estas cuentas sean tomadas como importantes en caso de que tengan muchos seguidores y no sigan a nadie.

Para visualizar el resto de la distribución, limitamos los ejes.

```
# Grado teniendo en cuenta gente following por un usuario
grado_g_out <- degree(g, mode='out')
grado_gDF_out <- as.data.frame(grado_g_out)
ggplot(grado_gDF_out, aes(grado_gDF_out)) + geom_histogram(binwidth = 1) +
  xlab('Vertex Degree') + ylab('Frequency') + ggttitle('Histogram of output degree') +
  xlim(-1,40) + ylim(-1,300)

## Warning: Removed 292 rows containing non-finite values (stat_bin).
## Warning: Removed 1 rows containing missing values (geom_bar).
```

Histogram of output degree



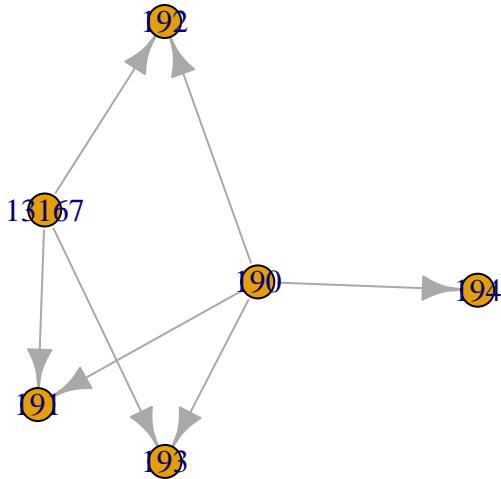
Como observamos en este otro histograma, la tendencia de las cuentas de este grafo es la de seguir a no más de 50 cuentas cada una.

4. Si seleccionamos la comunidad formada por los usuarios 190, 191, 192, 193, 194 y 13167, ¿podemos considerar que los usuarios que la forman están fuertemente conectados? ¿qué tipo de conexión tienen?

```
comunidad_1 <- c("190", "191", "192", "193", "194", "13167")
ig <- induced.subgraph(g, comunidad_1)
is.connected(ig, mode = 'strong')

## [1] FALSE
is.connected(ig, mode = 'weak')

## [1] TRUE
plot(ig)
```



La comunidad elegida no es fuertemente conexo. Esto quiere decir que, teniendo en cuenta la dirección de las conexiones, desde cualquier nodo no podemos llegar a cualquier otro.

Por el contrario, la comunidad seleccionada si que es débilmente conexa. Esto quiere decir que sin tener en cuenta las direcciones de las conexiones, desde cualquier nodo podemos llegar a cualquier otro. En este caso, podemos decir que entre los usuarios de esta comunidad tienen gente en común, ya sean followers o personas following.

5. ¿Qué usuario tiene más seguidores? ¿Qué usuario sigue a un mayor número de usuarios?

```
user_most_foll <- V(g)$name[grado_g_in==max(grado_g_in)]
```

El usuario 36 es el que tiene más seguidores en Twitter con 57 seguidores.

```
user_most_fwng <- V(g)$name[grado_g_out==max(grado_g_out)]
```

El usuario 11824 es el que sigue a un mayor número de usuarios en Twitter con un total de 238

6. Indica quiénes son los seguidores del usuario 1305.

```
user <- 1305
v <- V(g)$name[which(V(g)$name==user)]
neigs <- neighbors(g,v,mode = 'in')
cat('El usuario ', user, ' tiene un total de ', degree(g,v,mode='in'),
    'seguidores. Son los usuarios: ', neigs, "\n")
```

```
## El usuario 1305 tiene un total de 4 seguidores. Son los usuarios: 37 249 482 623
```

7. Indica a quién sigue el usuario 1373.

```
user <- 1373
v <- V(g)$name[which(V(g)$name==user)]
neigs <- neighbors(g,v,mode = 'out')
cat('El usuario ', user, ' sigue a un total de ', degree(g,v,mode='out'),
    'usuarios\n')
```

```
## El usuario 1373 sigue a un total de 57 usuarios
```

8. ¿Cuál es el ratio seguidores/siguiendo del usuario 13815, se puede considerar que es un usuario influyente?

```
ratio <- function(graph, v){
  followers <- degree(g,v,mode='in')
  following <- degree(g,v,mode='out')
```

```

ratio_sol <- followers/following
return(ratio_sol)
}
user <- 13815
v <- V(g)$name[which(V(g)$name==user)]
ratio_sol <- ratio(g,v)
cat('El ratio seguidores/siguiendo del usuario ', user, ' es de ', ratio_sol)

## El ratio seguidores/siguiendo del usuario 13815 es de 0.1111111

```

Si consideramos que un usuario es influyente cuando el número de seguidores es mucho mayor que el de personas siguiendo, el usuario 13815 con un ratio de 0.1111111 en el cociente seguidores/siguiendo no lo podemos considerar como influyente.

9. ¿Podemos decir que el grafo representa una gran comunidad? ¿O existen comunidades aisladas? Indica la comunidad o comunidades que encuentras en el grafo.

```

#eb <- edge.betweenness.community(g, directed = TRUE)
#sg <- spinglass.community(g) # Cannot work with unconnected graph
im <- infomap.community(g)
sizes(im)

## Community sizes
##   1   2   3   4   5   6   7   8   9   10  11  12
## 22283 74  52  40  35  34  33  31  31  30  30  28
##   13  14  15  16  17  18  19  20  21  22  23  24
##   28  27  26  24  22  21  20  17  18  17  15  15
##   25  26  27  28  29  30  31  32  33  34  35  36
##   15  15  14  14  14  14  14  13  12  12  12  10
##   37  38  39  40  41  42  43  44  45  46  47  48
##   11  11  10  10  10  10  9   9   9   8   8   7
##   49  50  51  52  53  54  55  56  57  58  59  60
##   7   7   6   6   6   6   6   3   5   5   5   5
##   61  62  63  64  65  66  67  68  69  70  71  72
##   4   4   4   3   3   3   3   3   2   2   2   2
##   73  74  75  76  77  78  79  80  81  82  83  84
##   2   2   2   2   2   2   2   2   2   2   2   2
##   85  86  87  88  89  90  91  92  93  94  95
##   2   2   2   2   2   2   2   2   2   2   2
im_modularity <- im$modularity

```

Para detectar comunidades existen diferentes algoritmos implementados en r. Dado que tenemos un grafo dirigido, el abanico de posibilidades se reduce a 3 (edge.betweenness.community, spinglass.community e infomap.community). Debido a la gran cantidad de nodos y aristas presentes en el grafo, el algoritmo que calcula el edge.betweenness en cada iteración lleva un largo periodo de procesamiento. Por otro lado, el algoritmo spinglass nos indica que no puede trabajar con grafos no conectados, lo que nos hace pensar que hay nodos que no tienen conexiones hacia ellos y por lo tanto se consideran como no conectados con el resto, ya que no son alcanzables desde otros nodos.

Dicho esto, utilizamos el algoritmo de infomap para la detección de comunidades, pese a que el valor de la modularidad sea muy bajo (0.0608574).

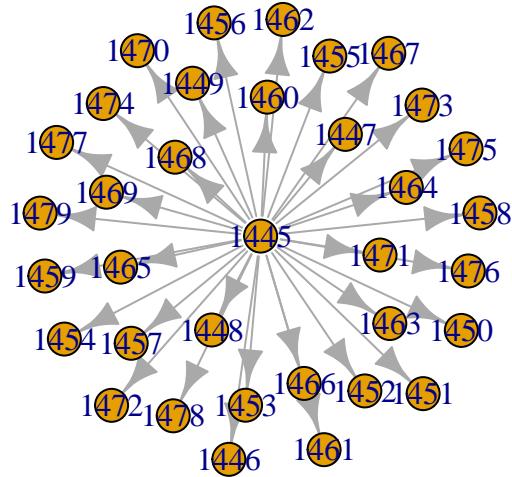
Se han detectado un total de 95 comunidades, de las cuales, existe una predominante formada por 22283 nodos.

Si comparamos el tamaño de la comunidad con mayor número de nodos, en comparación con el resto, podemos observar que la diferencia es muy grande. Esto puede dar lugar a consideraciones del tipo de que todo el grafo

se compone de una gran comunidad y que el resto de nodos han quedado agrupados en minicomunidades que a priori no serán de gran relevancia para el análisis de características en las comunidades.

10. Si tenemos en cuenta la comunidad de 35 usuarios del grafo, ¿cuál es el usuario menos importante? ¿Por qué?

```
size <- 35
size_cond <- which(sizes(im)==size)
com_35 <- induced.subgraph(g, which(membership(im) == size_cond))
plot(com_35)
```



Seleccionamos la comunidad con 35 integrantes y la dibujamos. Como se puede observar, está formada por un usuario que sigue al resto y por muchos usuarios que ninguno le sigue a él. Esto nos hace pensar que en esta comunidad, si medimos la importancia de un nodo como el ratio entre followers/following, este nodo central será el menos importante de la comunidad.

```
curr_node <- 0
curr_rat <- 0
for(n in 1:length(V(com_35))){
  node <- V(com_35)$name[[n]]
  rat <- ratio(com_35,node)
  if (rat <= curr_rat & rat != Inf) {
    curr_node <- node
    curr_rat <- rat
  }
  cat('Nodo: ',node,' con ratio: ', rat, '\n')
}
```

```
## Nodo: 1445 con ratio: 0
## Nodo: 1446 con ratio: Inf
## Nodo: 1447 con ratio: Inf
## Nodo: 1448 con ratio: Inf
## Nodo: 1449 con ratio: Inf
## Nodo: 1450 con ratio: Inf
## Nodo: 1451 con ratio: Inf
## Nodo: 1452 con ratio: Inf
## Nodo: 1453 con ratio: Inf
## Nodo: 1454 con ratio: Inf
## Nodo: 1455 con ratio: Inf
## Nodo: 1456 con ratio: Inf
## Nodo: 1457 con ratio: Inf
```

```

## Nodo: 1458 con ratio: Inf
## Nodo: 1459 con ratio: Inf
## Nodo: 1460 con ratio: Inf
## Nodo: 1461 con ratio: Inf
## Nodo: 1462 con ratio: Inf
## Nodo: 1463 con ratio: Inf
## Nodo: 1464 con ratio: Inf
## Nodo: 1465 con ratio: Inf
## Nodo: 1466 con ratio: Inf
## Nodo: 1467 con ratio: Inf
## Nodo: 1468 con ratio: Inf
## Nodo: 1469 con ratio: Inf
## Nodo: 1470 con ratio: Inf
## Nodo: 1471 con ratio: Inf
## Nodo: 1472 con ratio: Inf
## Nodo: 1473 con ratio: Inf
## Nodo: 1474 con ratio: Inf
## Nodo: 1475 con ratio: Inf
## Nodo: 1476 con ratio: Inf
## Nodo: 1477 con ratio: Inf
## Nodo: 1478 con ratio: Inf
## Nodo: 1479 con ratio: Inf

```

En el cálculo del ratio para cada nodo podemos observar dos tipos de resultados. El primero de ellos es el Inf. Este valor se obtiene cuando un nodo tiene followers pero no sigue a nadie. Se puede considerar como un nodo importante. El segundo resultado es el 0. Esto indica que el usuario no tiene followers y sigue a otros usuarios.

Por lo tanto y como se había comentado anteriormente mediante la representación gráfica, el nodo menos importante es el 1445 con un ratio de 0.

Otra forma de medir la importancia de un nodo es calculando el vector de autovalores. Esto da la idea de que un nodo es importante si los nodos que le apuntan son importantes. En este caso, no podemos aplicar este algoritmo debido a que es dirigido y sin ciclos, por lo que el vector será todo ceros.

11. Si ahora analizamos la comunidad más grande del grafo

```

size_cond <- which.max(sizes(im))
com_max <- induced.subgraph(g, which(membership(im) == size_cond))

```

a. ¿Cuál es el vértice más cercano al resto?

```

node_max_clos <- V(com_max)$name[which.max(closeness(com_max, normalized = TRUE,
                                                       mode = 'all'))]

```

Si seguimos el criterio de que un nodo es central si esta muy cerca del resto de nodos, podemos usar el algoritmo closeness que mide la centralidad de un nodo calculando la cercanía con el resto. Debido a esto, calculamos el nodo que tiene un máximo closeness y obtendremos el nodo más cercano al resto. Este nodo es el 897.

b. ¿Qué vértice es el que controla un mayor flujo de información?

```

node_max_bet <- V(com_max)$name[which.max(betweenness(com_max,
                                                       normalized = TRUE, directed = TRUE))]

```

Si seguimos el criterio de que un nodo es central si pasan muchos caminos por éste, podemos usar el algoritmo betweenness que mide la centralidad de un nodo calculando los caminos entre nodos que pasan por cada éste. Debido a esto, calculamos el nodo que tiene un máximo betweenness y obtendremos el nodo que controla un mayor flujo de información (más información pasa por este nodo). Este nodo es el 1368.