# Assignment 4&5 18230229

*Sai Krishna Lakshminarayanan*

*15 April 2019*

## Introduction

The assignment is done in order to make use of graphical analysis on the text domain. The book considered for this purpose is Strange Case of Dr Jekyll and Mr Hyde. It is extracted from the Project Guttenberg archive and separated into text files based on 10 chapters present. This is considered to be a story about a London legal person named Utterson who is investigating the strange split personality occurrences between his friend Dr.Jekyll and Edward.

## 2. Loading the book

Now, all the required packages are loaded into the markdown file.

```
library(tidyverse)      # data manipulation & plotting
library(stringr)        # text cleaning and regular expressions
library(tidytext)       # provides additional text mining functions
library(widyr)
library(igraph)
library(ggraph)
library(igraphdata)
library(kableExtra)
```

The book is manually stored in 10 text files split based on their chapters present. It is stored in the folder name based on the book name. Now, those details are noted down in order to call the text present in each and every chapter later when needed.

```
book_title = "thestrangecaseofdrjekyllandmrhyde"
chapters = 1:10
chapter_stem = "thestrangecaseofdrjekyllandmrhyde"
ext <-".txt"
folder<- "./thestrangecaseofdrjekyllandmrhyde/"

book <- tibble()
```

Now, a for loop is used inorder to parse through all the files and store the words that are present in a column. The dataframe book has 3 columns namely the book name denoting the name of the book, chapter denoting the chapter and finally word for the word that is present.

```
#read in each chapter
for(i in chapters){

  chapter <-paste0(folder,chapter_stem,i,ext)

  raw<-readChar(chapter, file.info(chapter)$size)

  chapter_text <- raw %>%
    gsub("[\r\n]+", " ", .) %>%
    gsub('^"',"",.) %>%
    gsub('"$',"",.)

  #creates a tibble with 3 cols: book_title, chapter, word
  words <- tibble(title = book_title, chapter=i, text = chapter_text) %>%
    unnest_tokens(word, text) %>% # tokenise the text
    filter(!word %in% stop_words$word) # remove stop words

  book <- rbind(book, words) # add rows to the book tibble

}
```

Now, based on this, the word correlation is obtained between the words. This is done by setting up minimum number of word pair limit and correlation limit. Once this is obtained, then the graph measures are found out and the correlation graph is obtained.

```
word_cor <- book %>%
  group_by(word) %>%
  filter(n() >= 12) %>% # minimum number of word pairs to consider; determines the number of nod
es; lower value -> more nodes
  pairwise_cor(item=word, feature=chapter) %>%
  filter(!is.na(correlation), correlation >= 0.60) # minimum correlation; determines the number
 of edges

correlation_graph <-graph_from_data_frame(word_cor,directed = FALSE)
```

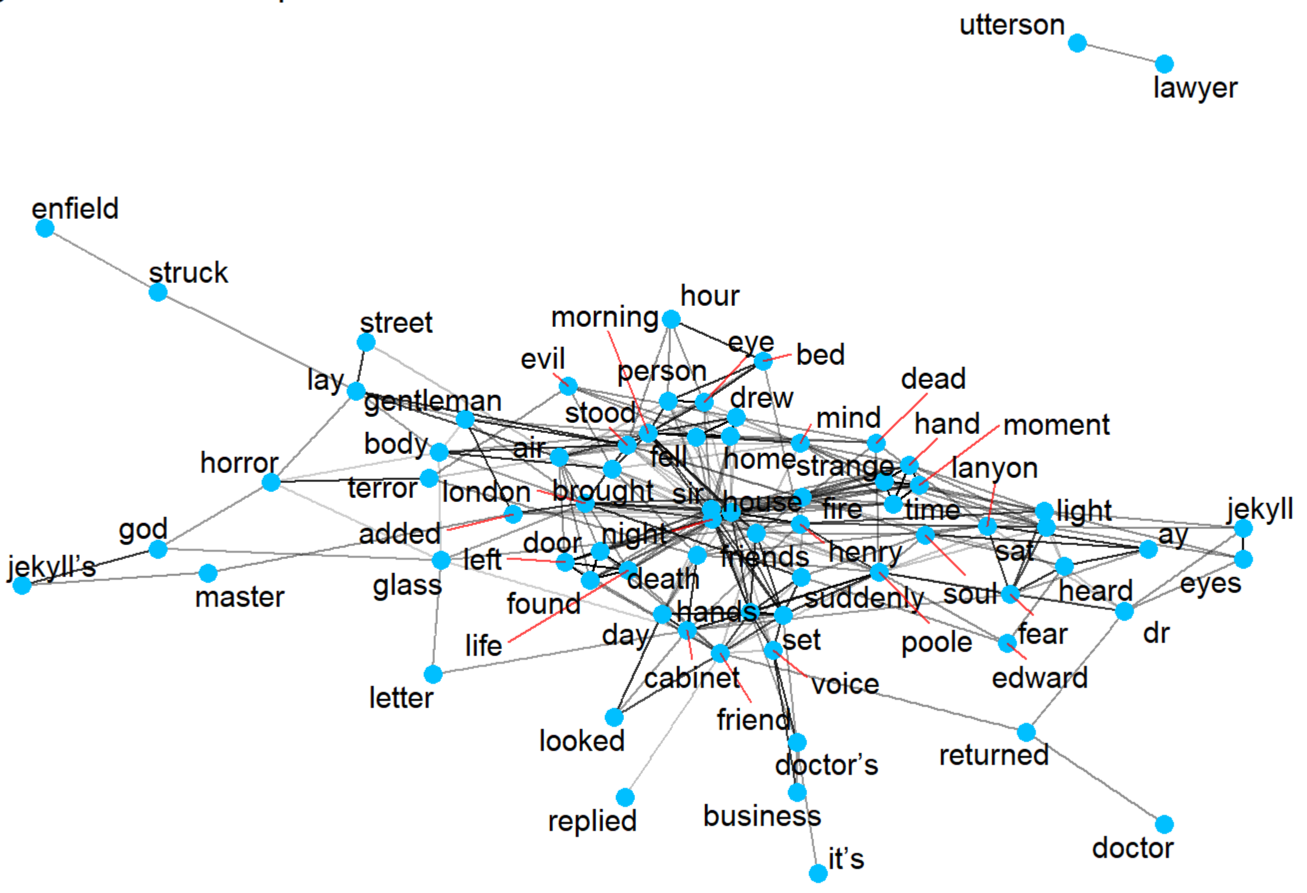
## 3. Graph Measurements

## Correlation Graph

Now, the correlation graph is obtained through the help of the ggraph() function. It can be done in two ways namely by using geom_node_text() and geom_node_label(). The major difference between the two is that text just produces only the text of the word present and label presents in a much more polished box way.

```
  ggraph(correlation_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_text(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alpha
= 0.7, segment.size = 0.4) +
  labs(title ="Fig 1 Correlation Graph of the book ")+# repel =TRUE stops labels overlapping; bu
t also  attaches a segment line between node and text, which may be visually confusing if it ove
rlaps with graph edges and nodes
  theme_void()
```

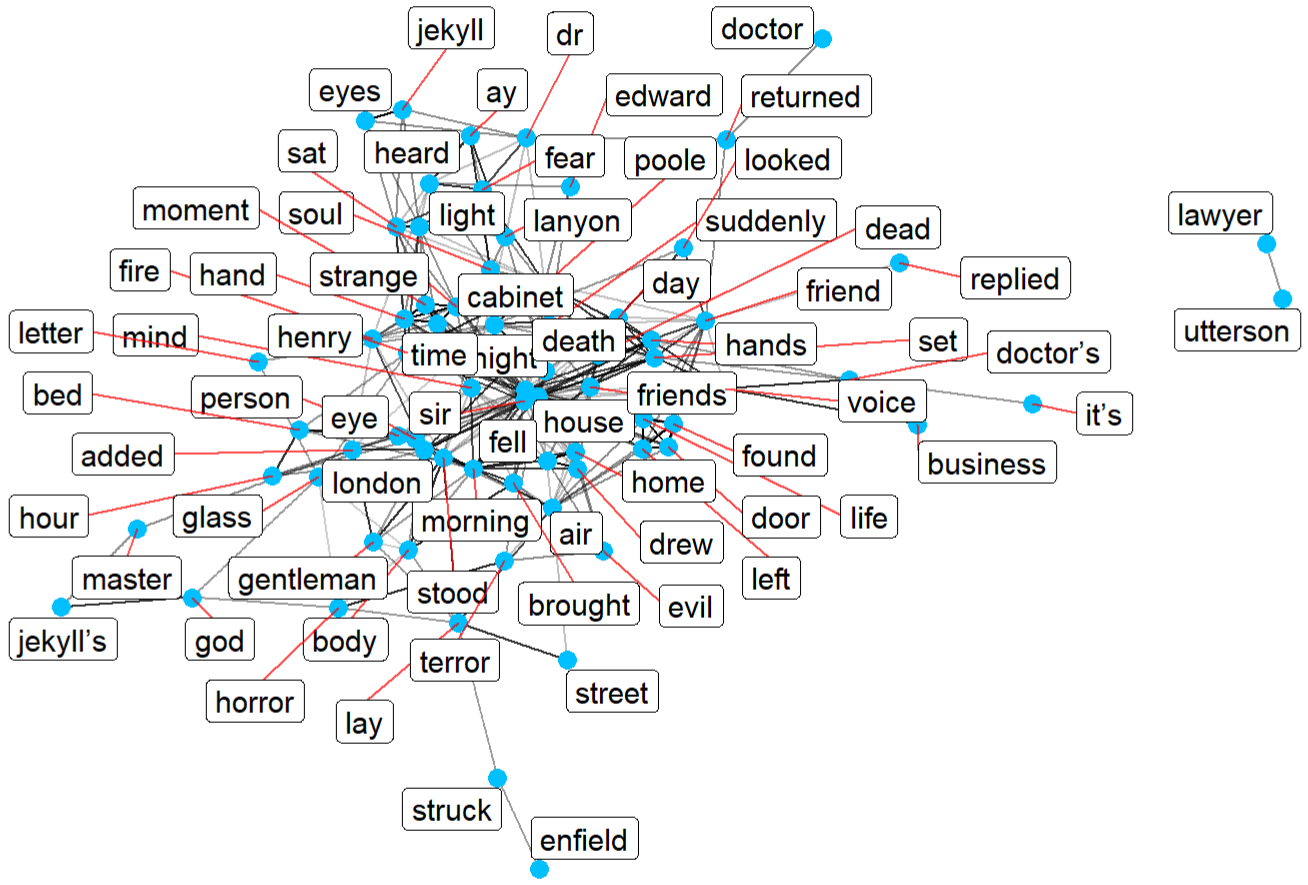
Fig 1 Correlation Graph of the book



Now, the same is done by using geom_node_label()

```
  ggraph(correlation_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_label(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alph
a = 0.7, segment.size = 0.4) +
  labs(title ="Fig 2 Correlation Graph of the book ")+# repel =TRUE stops labels overlapping; bu
t also  attaches a segment line between node and text, which may be visually confusing if it ove
rlaps with graph edges and nodes
  theme_void()
```

Fig 2 Correlation Graph of the book



From

these fraphs, the outline of the correlation between the words are obtained. But this alone is meaningless in order to understand the flowthrough of the plot story in the book. Other graph measures are to be found out in order to know the words which are important and carry forward throughout the book and words which are prevelant in a particular section only.

Now, calculating the diameter ,radius, degrees and eccentricity

```
print('The diameter of the graph is')
```

```
## [1] "The diameter of the graph is"
```

```
print(diameter(correlation_graph))
```

```
## [1] 7
```

```
print("The radius of the graph is")
```

```
## [1] "The radius of the graph is"
```

```
print(radius(correlation_graph))
```

```
## [1] 1
```

```
print("The degrees are")
```

```
## [1] "The degrees are"
```

```
print(degree(correlation_graph))
```

```
##    lawyer  utterson       air      mind   morning       sir     night 
##         2         2        32        28        32        56        56 
##    person     house      hour       bed      life    friend      left 
##        18        56         8        10        16        28        16 
##      door     found   friends       set     voice  doctor’s       day 
##        16        16        24        24        16        10        18 
##  suddenly     poole   cabinet     hands      dead      soul     death 
##        20        32        20        24        12        14        14 
##      fire      hand    moment   strange       sat     light    struck 
##        18        20        20        20        24        24         4 
##    looked   replied  returned       lay     stood     added      body 
##         6         2         6        14        26        14        16 
## gentleman     glass    london       eye    street      home   brought 
##        12        14        20        18         4        20        18 
##      drew      fell    terror    horror      time     heard   enfield 
##        20        20         8        10        20        16         2 
##    lanyon    master  business    doctor        dr      evil     henry 
##        18         4         4         2        14        10        18 
##      it’s    edward    jekyll      fear      eyes       god        ay 
##         2         6         8        16         8         6         8 
##  jekyll’s    letter 
##         4         4
```

```
print("The eccentricity is")
```

```
## [1] "The eccentricity is"
```

```
print(eccentricity(correlation_graph))
```

```
##    lawyer  utterson       air      mind   morning       sir     night
##         1         1         5         5         4         4         4
##    person     house      hour       bed      life    friend      left
##         5         4         6         5         4         5         4
##      door     found   friends       set     voice  doctor’s       day
##         4         4         4         5         5         6         5
##  suddenly     poole   cabinet     hands      dead      soul     death
##         5         5         5         5         5         5         5
##      fire      hand    moment   strange       sat     light    struck
##         4         5         5         5         6         6         6
##    looked   replied  returned       lay     stood     added      body
##         6         6         6         5         5         5         5
## gentleman     glass    london       eye    street      home   brought
##         5         5         4         5         6         5         5
##      drew      fell    terror    horror      time     heard   enfield
##         5         5         5         6         5         5         7
##    lanyon    master  business    doctor        dr      evil     henry
##         5         6         6         7         6         6         5
##      it’s    edward    jekyll      fear      eyes       god        ay
##         7         6         7         6         7         6         5
##  jekyll’s    letter
##         7         5
```

From the values in the degree and eccentricity, we tend to get some more indepth idea into the prevelance of the words that are present in the book.

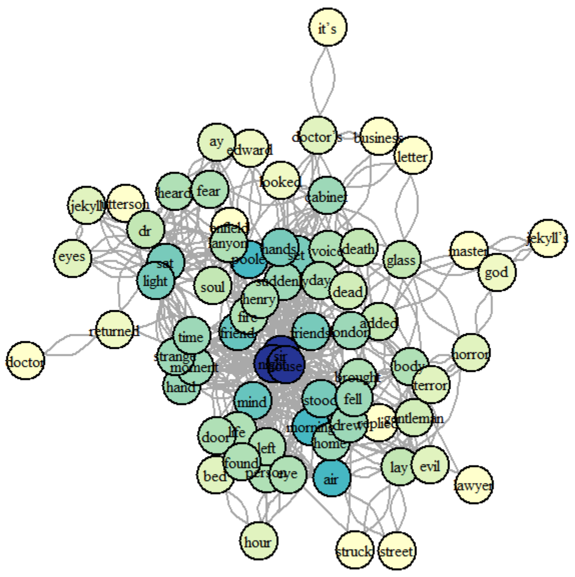
Based on these, we understand that the words with large centrality are present in the centre of the correlation graph whereas the ones with less centrality are at the outside. The diameter has represented the maximum eccentricity and the radius has done the minimum eccentricity present in the graph. So, we can say now from the values and graph that the outermost nodes have the highest eccentricity.

## Centrality

The degree centrality measure is used to denote the number of edges that are connected to a node. Since it is undirected, no separate in and out degree are considered. It helps in understanding the degrees of the nodes that are present. Based on this, the words with large and small degrees can be found out.

```
lyt <- layout_with_kk(correlation_graph)
ylGnBl5<-c("#FFFFCC","#C7E9B4","#7FCDBB","#40B6C4","#2C7FB8" ,"#253494")

#this gives you the colors you want for every point
ylOrBn<-colorRampPalette(ylGnBl5)
fine = 20

cent<- degree(correlation_graph)
graphCol = ylOrBn(fine)[as.numeric(cut(cent,breaks = fine))]
plot(correlation_graph,  vertex.color=graphCol, main="Fig 3.Degree Centrality", vertex.label.cex
=0.5,vertex.label.color="black", layout = -lyt[, 2:1],vertex.size=15)
```

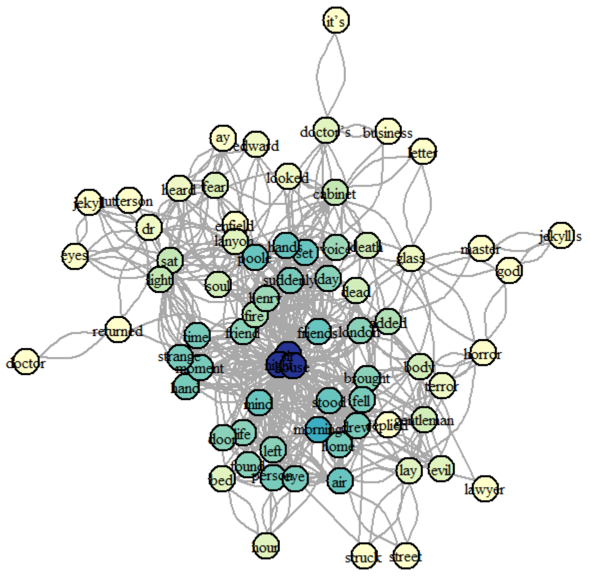
## Fig 3.Degree Centrality



Similarly the eigen centrality is found out. This helps to understand the influence of the node in the network. In this, we can find out how the words impact the story in the book,

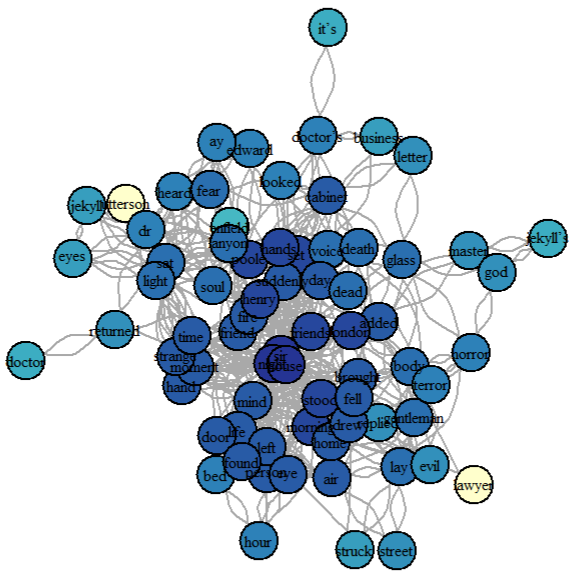
```
cent<- eigen_centrality(correlation_graph, weights=NA)$vector
graphCol = ylOrBn(fine)[as.numeric(cut(cent,breaks = fine))]
plot(correlation_graph,  vertex.color=graphCol, main="Fig 4 Eigen Centrality", vertex.label.cex=
0.5,vertex.label.color="black", layout = -lyt[, 2:1],vertex.size=10)
```

## Fig 4 Eigen Centrality



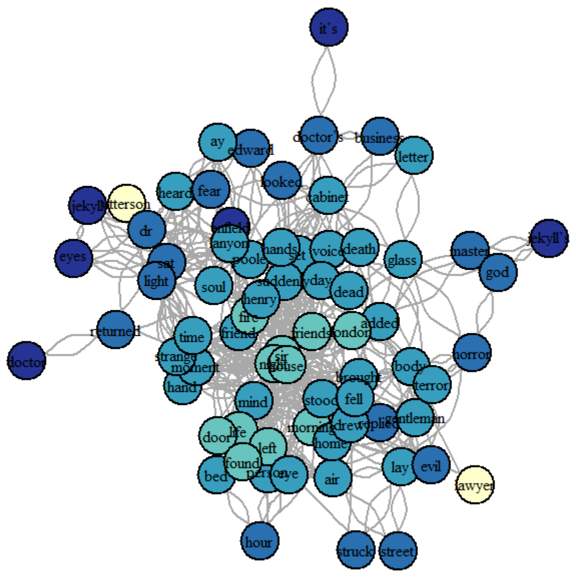
```
cent<- closeness(correlation_graph)
graphCol = ylOrBn(fine)[as.numeric(cut(cent,breaks = fine))]
plot(correlation_graph,  vertex.color=graphCol, main="Fig 5 Closness Centrality", vertex.label.c
ex=0.5,vertex.label.color="black", layout = -lyt[, 2:1],vertex.size=15)
```

## Fig 5 Closness Centrality



```
ecc<- eccentricity(correlation_graph)
graphCol = ylOrBn(fine)[as.numeric(cut(ecc,breaks = fine))]
plot(correlation_graph,  vertex.color=graphCol, main="Fig 6 Eccentricity", vertex.label.cex=0.5,
vertex.label.color="black", layout = -lyt[, 2:1],vertex.size=15)
```

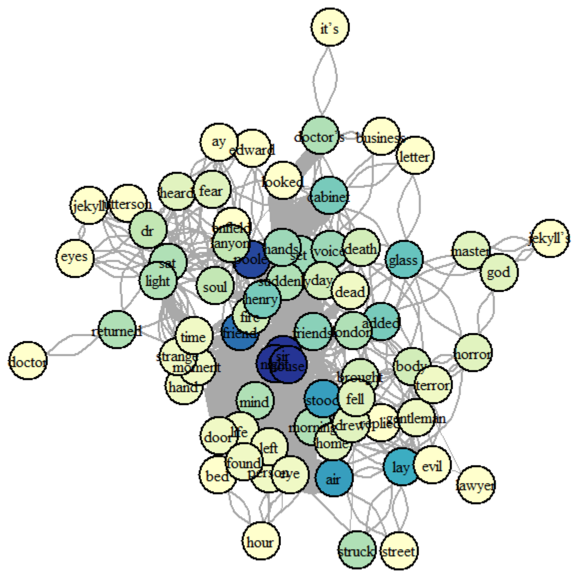
## Fig 6 Eccentricity



The betweeness centrality helps in measuring the shortest path among the paired nodes in a connected graph. A score is given based on these paths. Therefore, most frequent words in the shortest path will be having large betweeness centrality score.

```
cent <- betweenness(correlation_graph, weights=NA)
graphCol = ylOrBn(fine)[as.numeric(cut(cent,breaks = fine))]
plot(correlation_graph,  vertex.color=graphCol, main="Fig 7 Betweenness Centrality", vertex.labe
l.cex=0.5,vertex.label.color="black", edge.width = cent/10, layout = -lyt[, 2:1],vertex.size=15)
```

Fig 7 Betweenness Centrality



## 4. Community Detection

The community detection has to be done to find the strong internally clusters which are also having only weak connections with other clusters. It will help us to disintegrate the book into several sub sections for better understanding of the plot and flowthrough present.

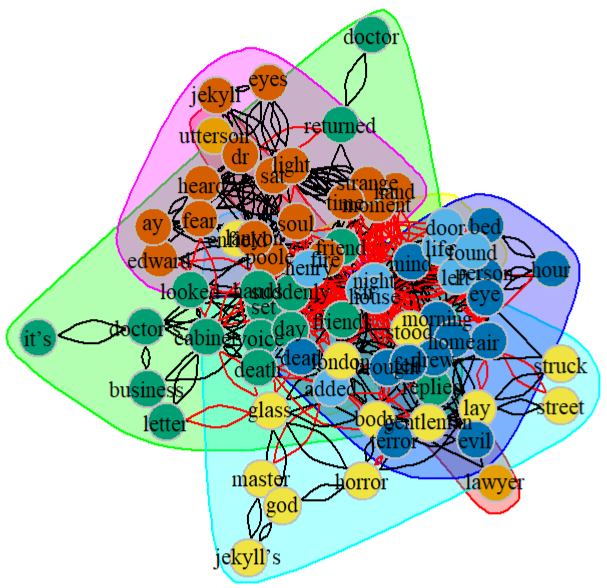
There are several community detection algorithms that are used to perform this task. It is important to execute with them and find out the best one that is suitable for our book that is present. The communities are to be considered as subgraph.For this purpose 3 algorithms are selected and the best among them is to be selected. They are, 1. Louvian 2. Walktrap 3. Edgebetweenness

### 4.1 Louvian

The louvian algorithm uses the concept of greedy optimization to optimise the modularity during the partitioning of network. It is a two step process with the initial one in which small communities are found by local optimization and the second one in which such small communities are merged into a larger one using modularity maximisation technique.

```
example.louv<- cluster_louvain(correlation_graph)
# membership vector
b<-membership(example.louv)


mod_max <- example.louv$modularity
k_opt <- length(example.louv)

# we can  use the plot.igraph approach to visualise the communities
plot(example.louv, correlation_graph, layout=lyt ,vertex.frame.color="gray", vertex.label.color=
"black", vertex.label.cex=0.7,main = paste("Fig 8. Louvain k=", max(k_opt), "; Modularity=", max
(round(mod_max,2))))
```

**Fig 8. Louvain k= 6 ; Modularity= 0.38**



From this, it is found that there are 6 communities and the moduarity is 0.38

```
source("my_modularity.R")
source("cluster_measurements.R")
intra.cluster.df<-cluster_measurements(correlation_graph,example.louv)
knitr::kable(intra.cluster.df, align = 'l', caption = paste0(algorithm(example.louv),  " : size,
modularity and density scores of communities detected"))
```

multi level : size, modularity and
density scores of communities
detected

| cluster | size | modularity | density |
|---|---|---|---|
| -1 | 72 | 0.0000000 | 0.2284820 |
| 1 | 2 | 0.0034129 | 2.0000000 |
| 2 | 9 | 0.0839158 | 1.5000000 |
| 3 | 17 | 0.1028805 | 0.5000000 |
| 4 | 13 | 0.0667456 | 0.5384615 |
| 5 | 14 | 0.1157229 | 0.8571429 |
| 6 | 17 | 0.1441523 | 0.7500000 |

Now the intra cluster values are found out and the modularity and density are noted. Here, it is seen that there is no misbehaviour or abnormality in the communities.

```
par(mfrow=c(1,1))
for(i in 1:length(unique(example.louv$membership))){
  plot(induced.subgraph(correlation_graph,example.louv$names[example.louv$membership==i]))
  title(paste0("Fig 9.",i,"Louvian based community ",i))
}
```

Fig 9.1Louvian based community 1

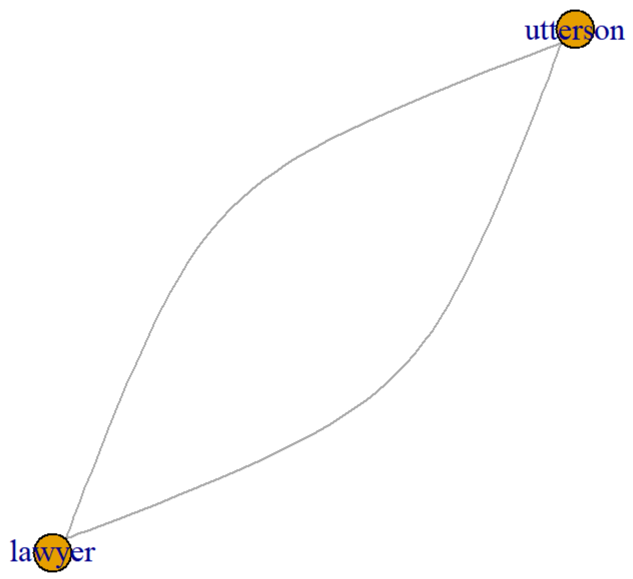
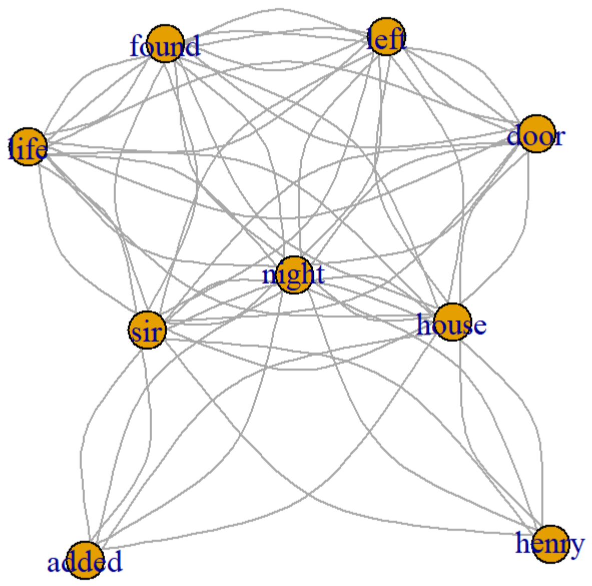
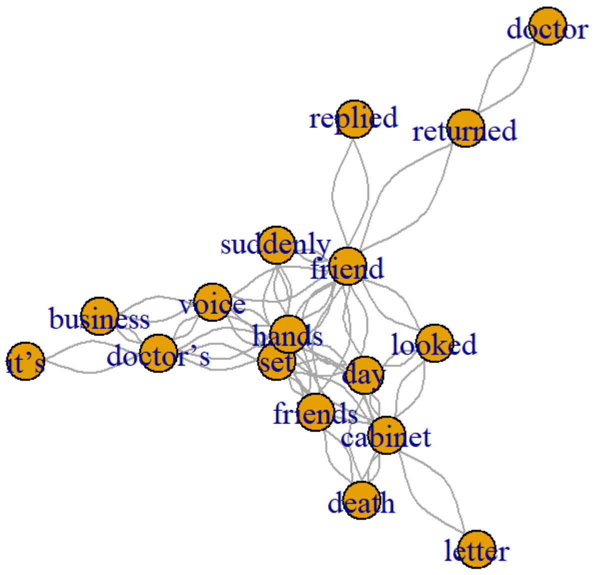


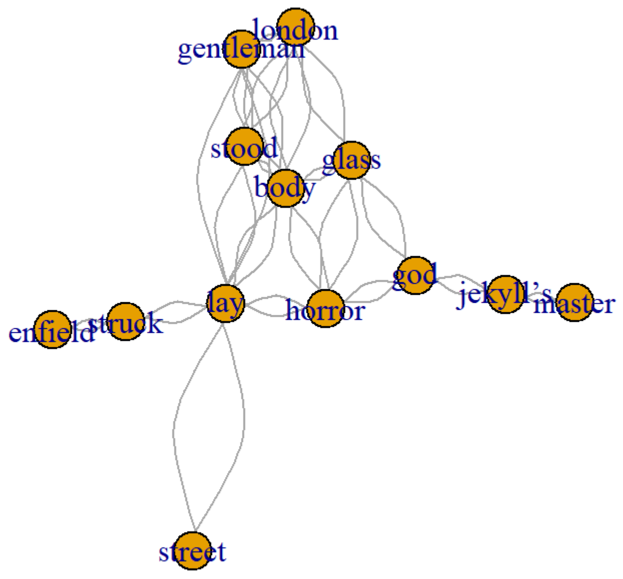
Fig 9.2Louvian based community 2



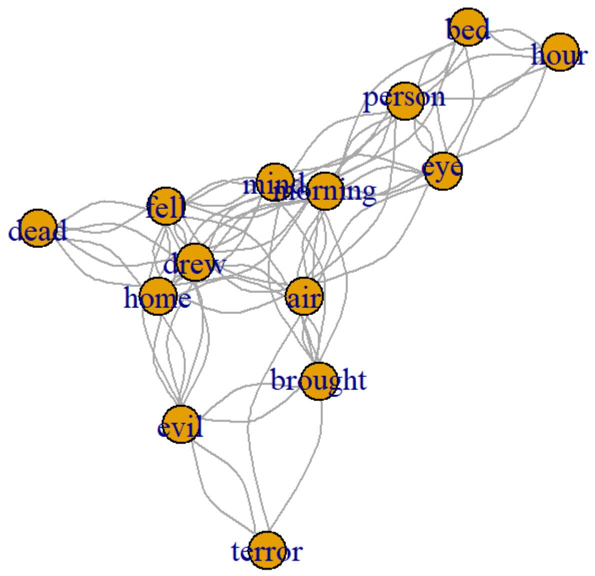
## Fig 9.3Louvian based community 3

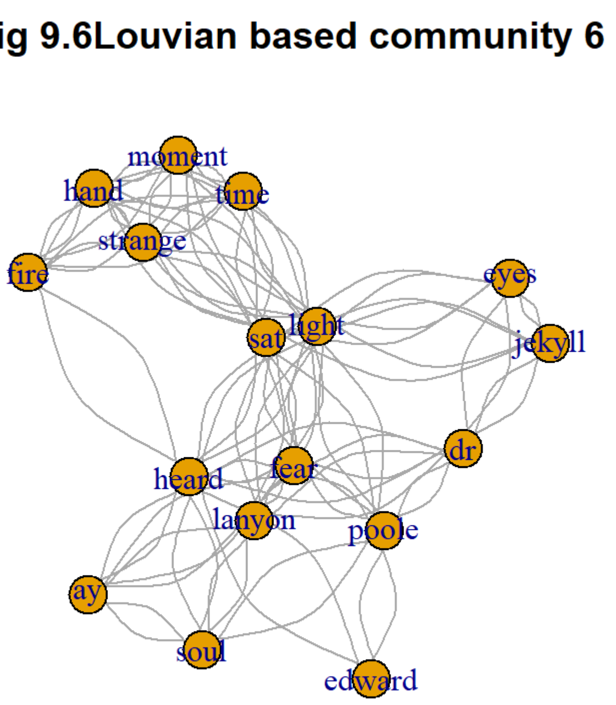


## Fig 9.4Louvian based community 4



## Fig 9.5Louvian based community 5





Now, each sub section is individually plotted in order to know about the intra cluster strength of each of them.

From community 1, it can be understood that mr.utterson is a lawyer

From community 2, it can be seen that the narration is about the happenings in henry home at night.

From community 3, it is seen that investigation of a death scene is being done.

From community 4, the transformation of dr.jekyll to mr.hyde and the havoc caused as a result is seen.

From community 5, the negative fallouts as a result of the incidents caused by the transformation is seen.

From community 6, the various characters namely jekyll ,hyde,poole ,edward etc and their final actions and emotions is visible.

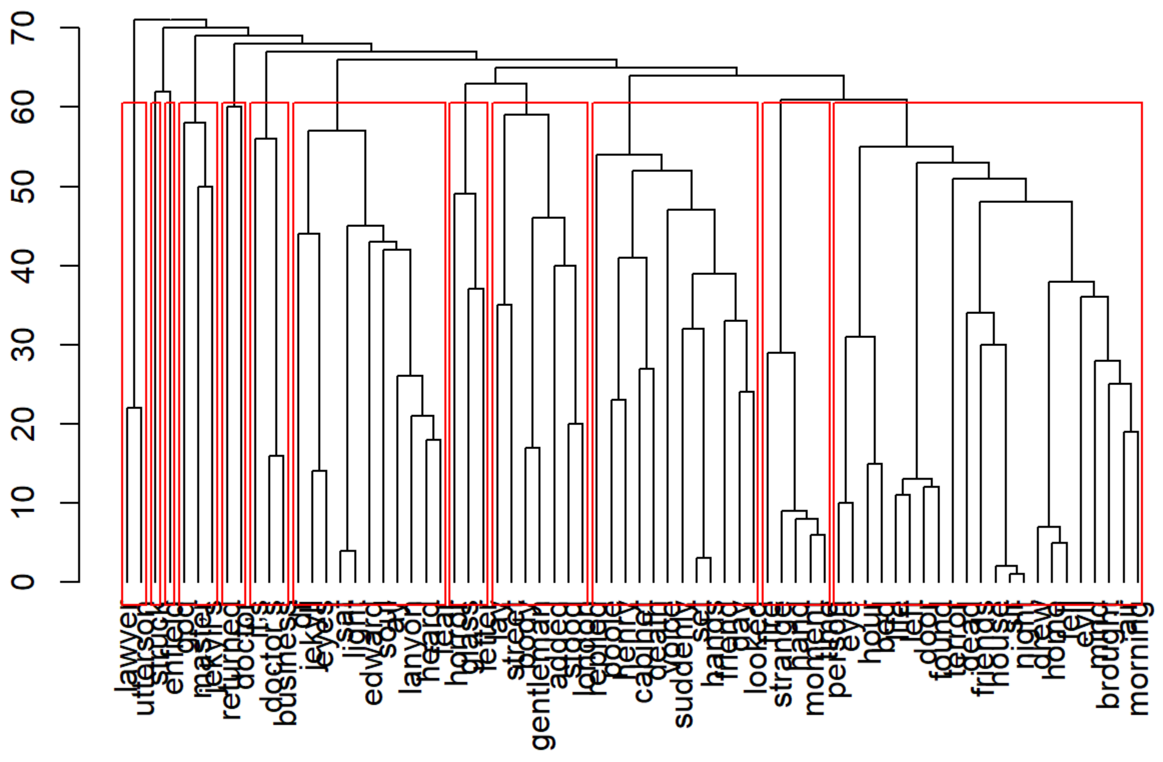
Thus, based on this louvian approach, a pretty clear picture is obtained.

## 4.2 Walktrap

The walktrap makes use of the concept of hierarchical clustering. It is based on the assumption that random walks which are short distanced tends to be in the same community. So, based on this approach, it slowly partitions every node and the merging occurs and the distanced are updated till the desired length is finished.

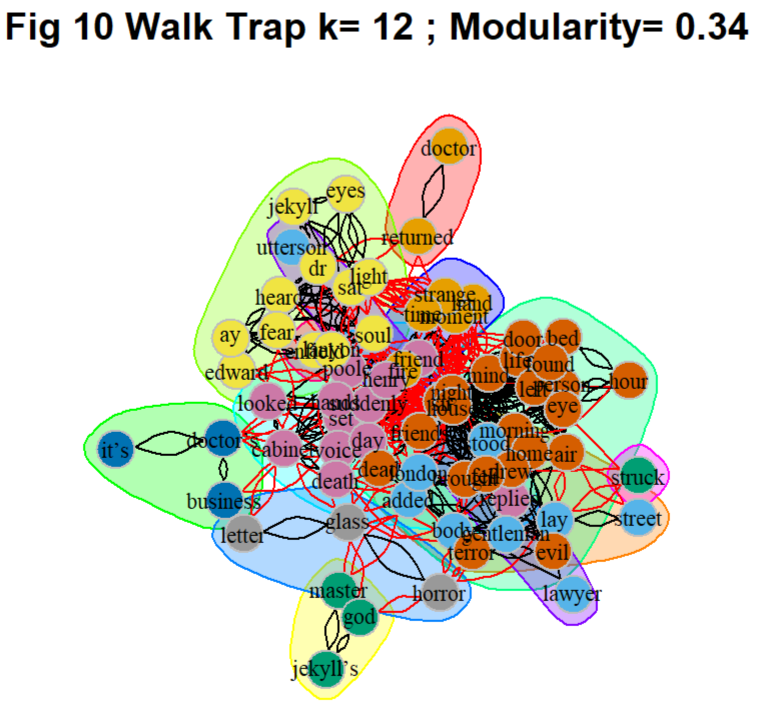
```
set.seed(3231)

g<-correlation_graph
example.wt<- cluster_walktrap(g,  membership = TRUE, merges = TRUE)

# in contrast to previous approach I use the communities object to get the max modularity
mod_max<- max(example.wt$modularity)

# returns the number of communities at the optimal modularity value of
k_opt<- length(example.wt)

# hclust object created from communities object
# necessary if I want to call rect.hclus to draw a cit
example.wt.hclust <- as.hclust(example.wt)

plot(as.dendrogram(example.wt.hclust))
title("Fig 10. Dendogram for Walktrap")

rect.hclust(example.wt.hclust, k = k_opt, border = "red")
```

Fig 10. Dendogram for Walktrap



From the dendogram, it is seen that the k value for the walktrap is very high when compared to louvian. Now, based on this k value, the communities are obtained and the moduarity is done.

```
# we can  use the plot.igraph approach to visualise the communities
plot(example.wt, g, layout=lyt,vertex.frame.color="gray", vertex.label.color="black", vertex.lab
el.cex=0.7,main = paste("Fig 10 Walk Trap k=", k_opt, "; Modularity=", round(mod_max,2)))
```



It is seen that the moduraliy for walktrap is lesser than that of the louvian and the communities are twice that of the louvian having 12 communities.

```
source("my_modularity.R")
source("cluster_measurements.R")
intra.cluster.df<- cluster_measurements(g,example.wt)
knitr::kable(intra.cluster.df, align = 'l', caption = paste0(algorithm(example.wt),  " : size, m
odularity and density scores of communities detected"))
```

walktrap : size, modularity and density scores of communities detected

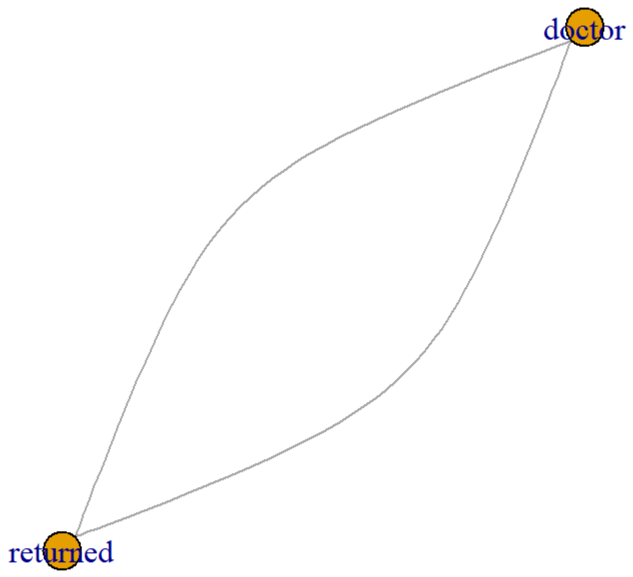
| cluster | size | modularity | density |
|---|---|---|---|
| -1 | 72 | 0.0000000 | 0.2284820 |
| 1 | 2 | 0.0034129 | 2.0000000 |
| 2 | 7 | 0.0394070 | 1.1428571 |
| 3 | 3 | 0.0068024 | 1.3333333 |
| 4 | 11 | 0.0811128 | 0.9454545 |
| 5 | 3 | 0.0068024 | 1.3333333 |

| cluster | size | modularity | density |
|---|---|---|---|
| 6 | 22 | 0.2158003 | 0.7965368 |
| 7 | 12 | 0.0866954 | 0.8484848 |
| 8 | 3 | 0.0068024 | 1.3333333 |
| 9 | 5 | 0.0330737 | 2.0000000 |
| 10 | 2 | 0.0034129 | 2.0000000 |
| 11 | 1 | 0.0000000 | NaN |
| 12 | 1 | 0.0000000 | NaN |

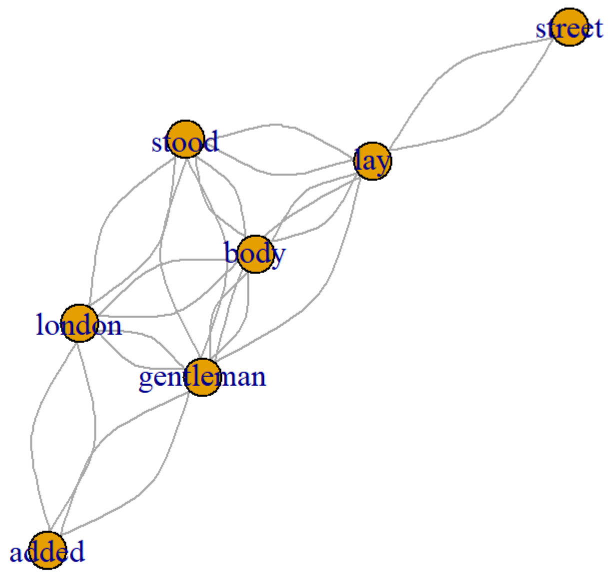
The modularity and intra cluster measurement are obtained. It is seen that there are two single node community present.

```
par(mfrow=c(1,1))
for(i in 1:length(unique(example.wt$membership))){
  plot(induced.subgraph(correlation_graph,example.wt$names[example.wt$membership==i]))
  title(paste0("Fig 11.",i,"Walktrap based community ",i))
}
```

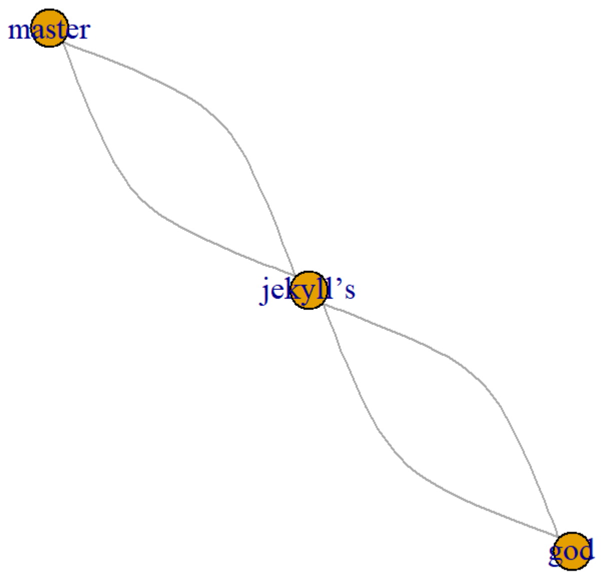
## Fig 11.1Walktrap based community 1



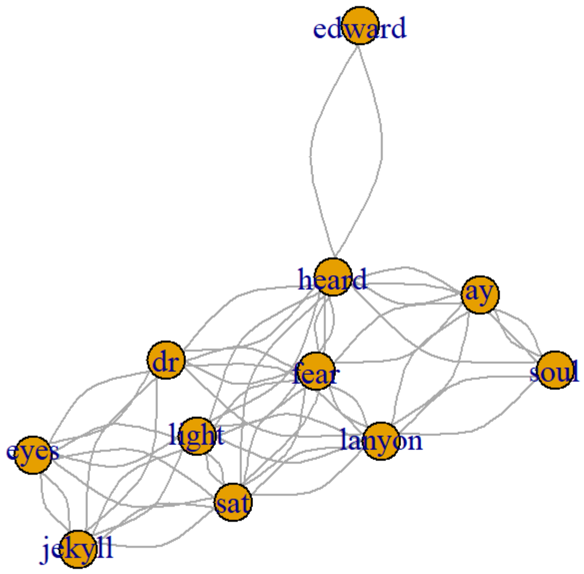
## Fig 11.2Walktrap based community 2



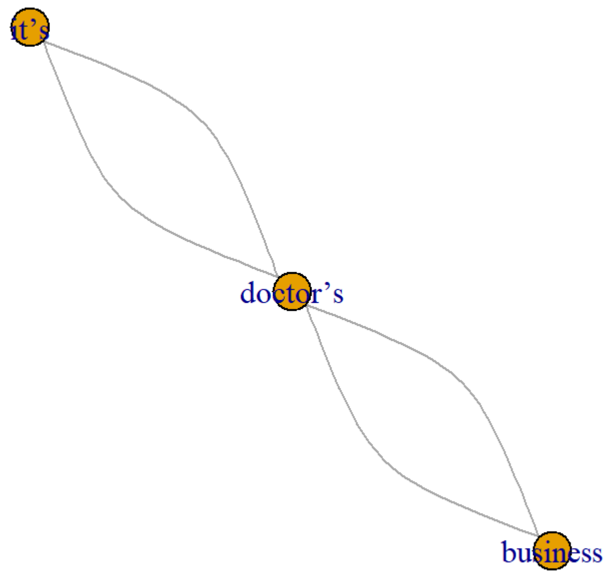
## Fig 11.3Walktrap based community 3



## Fig 11.4Walktrap based community 4



## Fig 11.5Walktrap based community 5



## Fig 11.6Walktrap based community 6

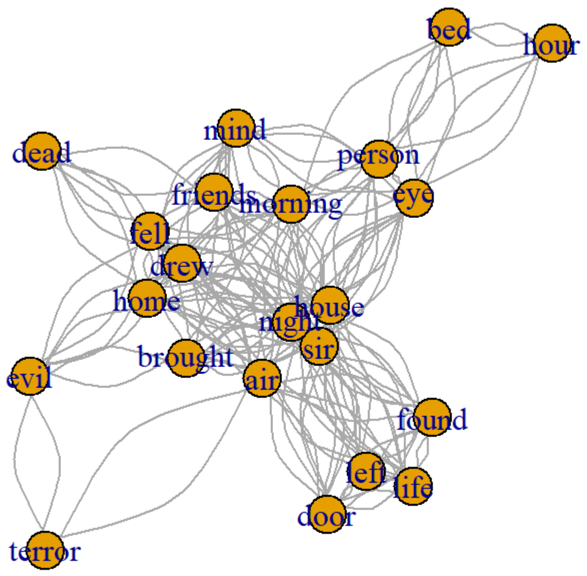


Fig 11.7Walktrap based community 7

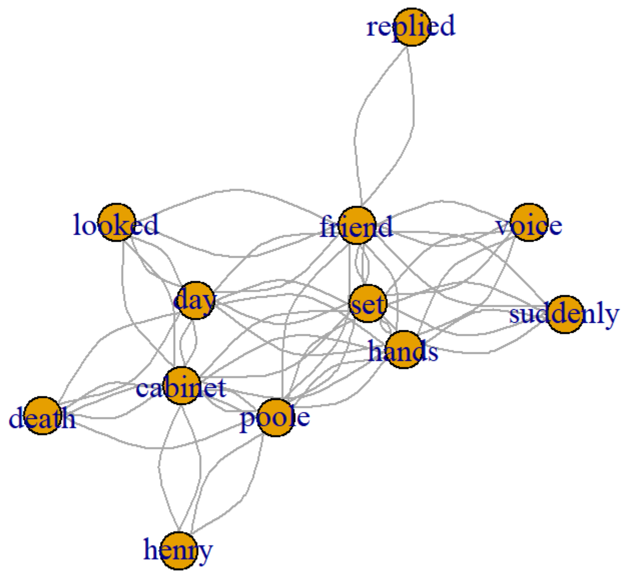
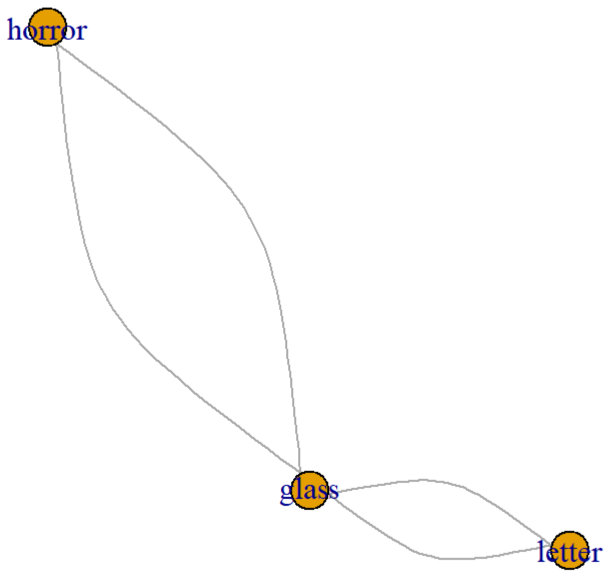
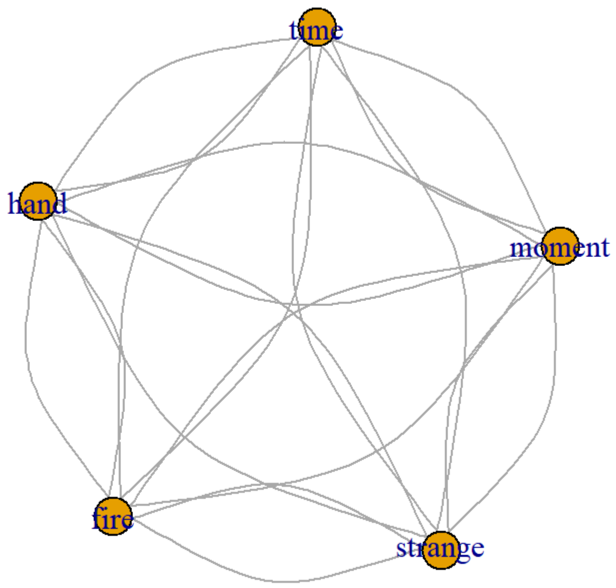


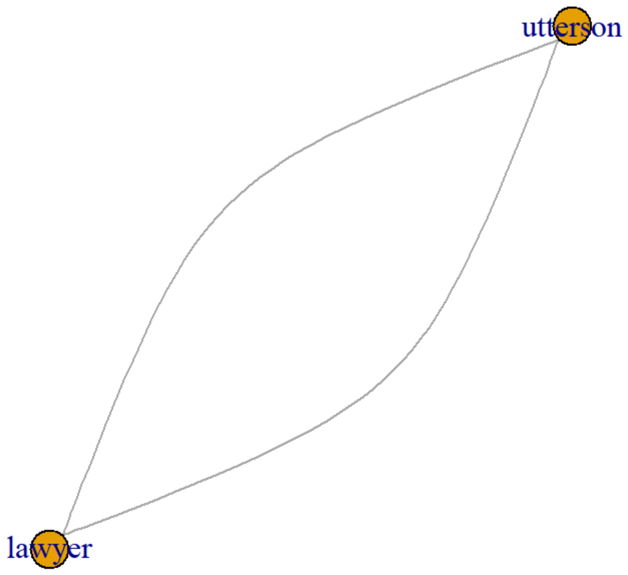
Fig 11.8Walktrap based community 8



## Fig 11.9Walktrap based community 9



## Fig 11.10Walktrap based community 10



## Fig 11.11Walktrap based community 11

## Fig 11.12Walktrap based community 12

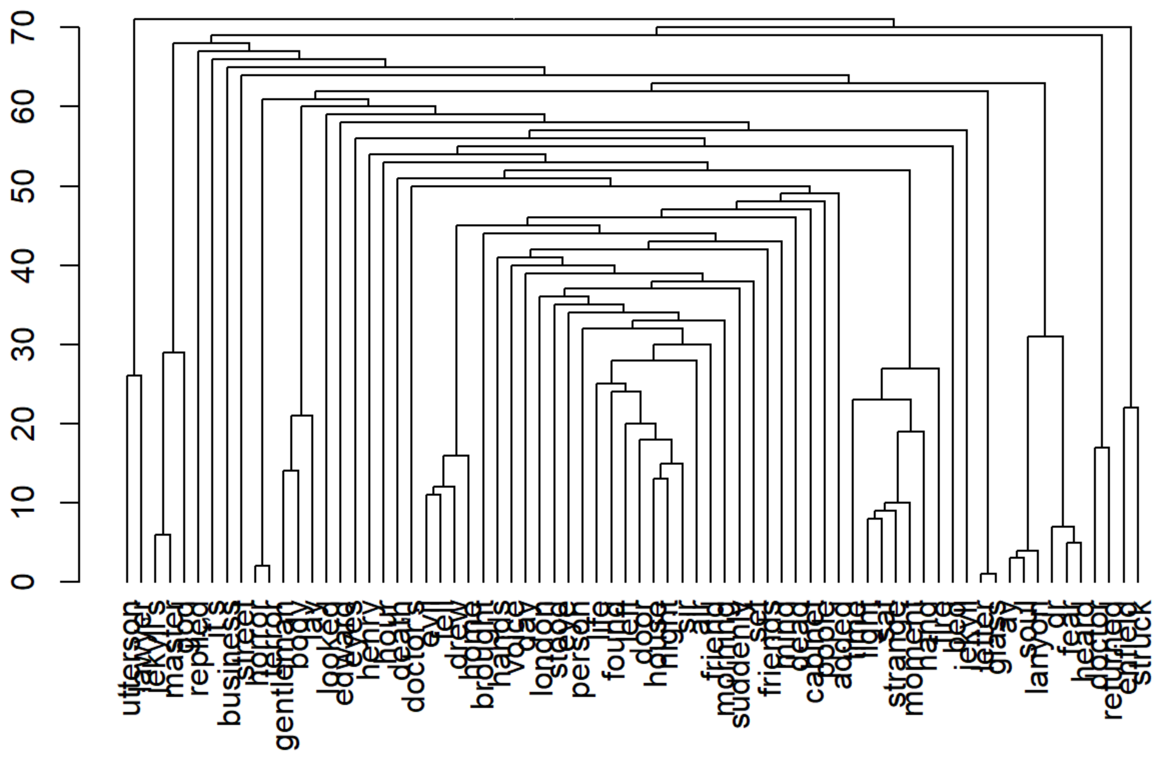
All the communities are displayed. Some of the are similar to the louvian and some of them are much more splitted to give separate details.

## 4.3 Edge betweeness

The edge betweeness community detection is done by removing edges continuously from the network. It then caculated the remaining connected components for the centrality. It focuses on the edges that are to be present most likely between the communities in the network.

```
set.seed(3233)
example.eb<- edge.betweenness.community(g, membership = TRUE, merges = TRUE)

# plots the dendogram of partitions
plot(as.dendrogram(example.eb))
title("Fig 12 Dendogram for edge betweeness")
```

**Fig 12 Dendogram for edge betweeness**

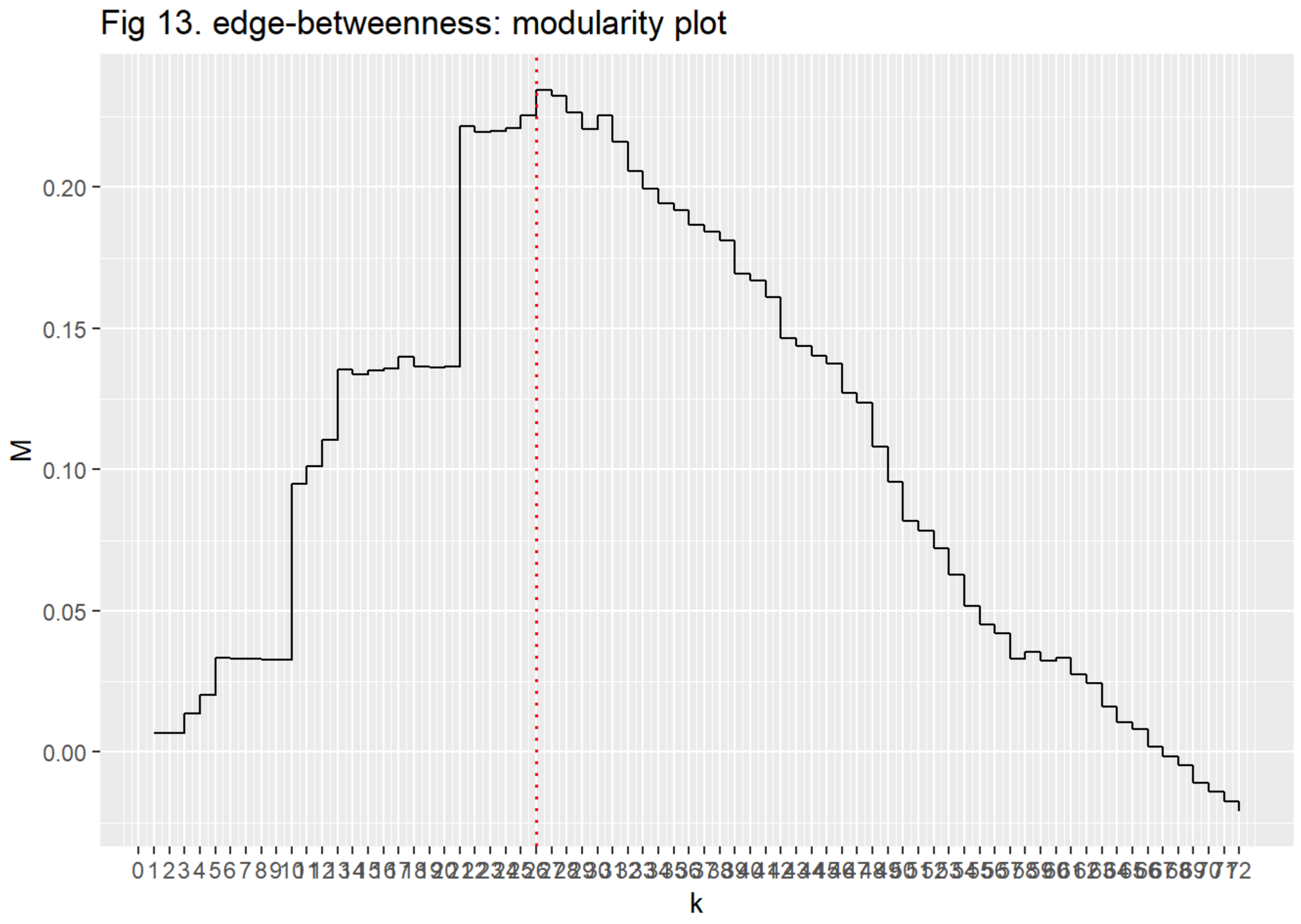


```
# 2 column matrix to record the k:modularity pairs
m<-matrix(nrow=length(V(g)), ncol=2)

# for each value of k make cut the dendogram
# measure the modularity at the cut

for(j in 1:length(V(g))){

  b <- cut_at(example.eb, no=j)
  mod<-modularity(g, b)

  m[j,1] = j
  m[j,2] = mod
}

# which value of k has the highest modularity
kmax<-m[,1][which((m[,2])==max(m[,2]))]

# make a data frame from m
m.df<- as.data.frame(m)
colnames(m.df) <- c("k","M")

n<-vcount(g)

# plot k vs Modularity
require(ggplot2)
```

From the dendogram, it is seen that the k value for edge betweeness is much larger than walktrap and louvian.

The k value is decided based on the modularity plot. The k value corresponding towards the maximum modurality value is taken. It is seen to be as 26.

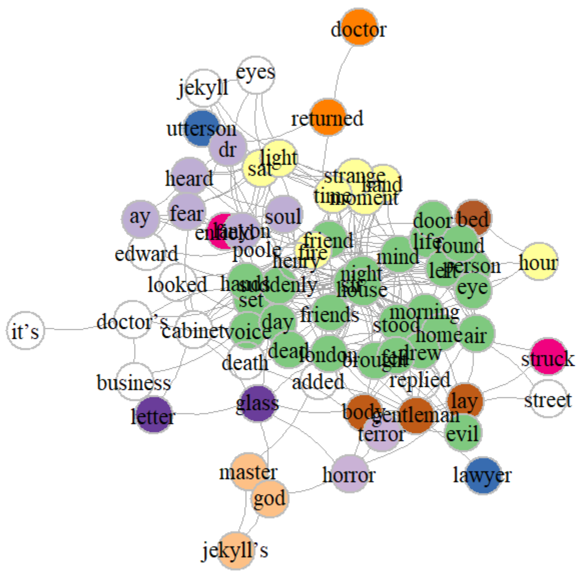
```
ggplot(m.df, aes(x=k,y=M)) + geom_step(direction="hv") +
  scale_x_continuous(breaks=c(0:n), labels=c(0:n)) +
   geom_vline(xintercept = kmax, linetype="dotted",
                color = "red", size=0.7) +
  ggtitle("Fig 13. edge-betweenness: modularity plot")
```



Now, the communities are plotted based on the k value and the modularity

```
k<-kmax
b <- cut_at(example.eb, no=kmax)
mod<-modularity(g, b)

colours <- c('#7fc97f','#beaed4','#fdc086','#ffff99','#386cb0','#f0027f','#bf5b17','#ff7f00','#c
ab2d6','#6a3d9a','#ffff99','#b15928')

  plot(g,layout = lyt, vertex.color=colours[b], vertex.frame.color="gray", vertex.label.color="b
lack", vertex.label.cex=0.7, edge.curved=0.2, edge.width=0.4, main = paste("Fig 14. Edge_Between
ness Comm., k=", k, "; Modularity=", round(mod,2)))
```

**Fig 14. Edge_Betweenness Comm., k= 26 ; Modularity= 0.23**



The modurality is lower than the others. Now, the size modularity and density scores of the communities are detected. It is seen that there are several single node community present

```
source("my_modularity.R")
source("cluster_measurements.R")
intra.cluster.df<- cluster_measurements(g,example.eb)

knitr::kable(intra.cluster.df, align = 'l', caption = paste0(algorithm(example.eb),  " : size, m
odularity and density scores of communities detected"))
```

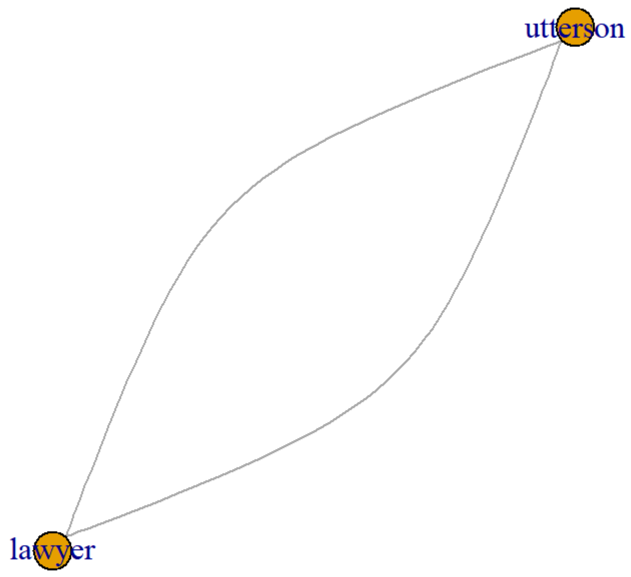
edge betweenness : size, modularity and density scores of communities detected

| cluster | size | modularity | density |
|---|---|---|---|
| -1 | 72 | 0.0000000 | 0.2284820 |
| 1 | 2 | 0.0034129 | 2.0000000 |
| 2 | 27 | 0.2466105 | 0.7350427 |
| 3 | 1 | 0.0000000 | NaN |
| 4 | 1 | 0.0000000 | NaN |
| 5 | 1 | 0.0000000 | NaN |
| 6 | 1 | 0.0000000 | NaN |
| 7 | 1 | 0.0000000 | NaN |
| 8 | 6 | 0.0362521 | 1.4666667 |
| 9 | 1 | 0.0000000 | NaN |

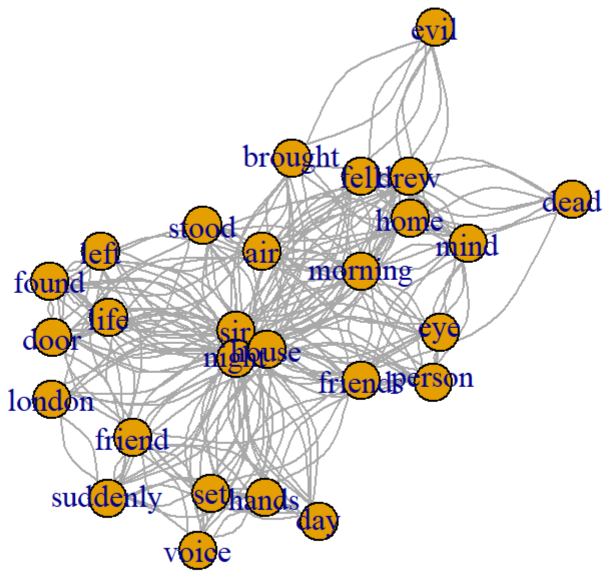
| cluster | size | modularity | density |
|---|---|---|---|
| 10 | 7 | 0.0608346 | 1.8095238 |
| 11 | 2 | 0.0034129 | 2.0000000 |
| 12 | 1 | 0.0000000 | NaN |
| 13 | 1 | 0.0000000 | NaN |
| 14 | 2 | 0.0034129 | 2.0000000 |
| 15 | 3 | 0.0101684 | 2.0000000 |
| 16 | 1 | 0.0000000 | NaN |
| 17 | 2 | 0.0034129 | 2.0000000 |
| 18 | 1 | 0.0000000 | NaN |
| 19 | 2 | 0.0034129 | 2.0000000 |
| 20 | 3 | 0.0068024 | 1.3333333 |
| 21 | 1 | 0.0000000 | NaN |
| 22 | 1 | 0.0000000 | NaN |
| 23 | 1 | 0.0000000 | NaN |
| 24 | 1 | 0.0000000 | NaN |
| 25 | 1 | 0.0000000 | NaN |
| 26 | 1 | 0.0000000 | NaN |

```
par(mfrow=c(1,1))
for(i in 1:length(unique(example.eb$membership))){
  plot(induced.subgraph(correlation_graph,example.eb$names[example.eb$membership==i]))
  title(paste0("Fig 15.",i,"edge betweeness based community ",i))
}
```

## Fig 15.1edge betweeness based community 1



## Fig 15.2edge betweeness based community 2



## Fig 15.3edge betweeness based community 3

## Fig 15.4edge betweeness based community 4

## Fig 15.5edge betweeness based community 5

## Fig 15.6edge betweeness based community 6

## Fig 15.7edge betweeness based community 7

## Fig 15.8edge betweeness based community 8

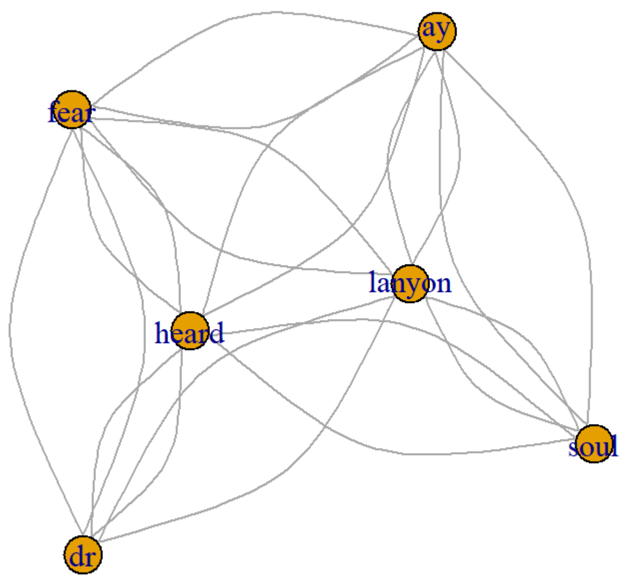


Fig 15.9edge betweeness based community 9
death

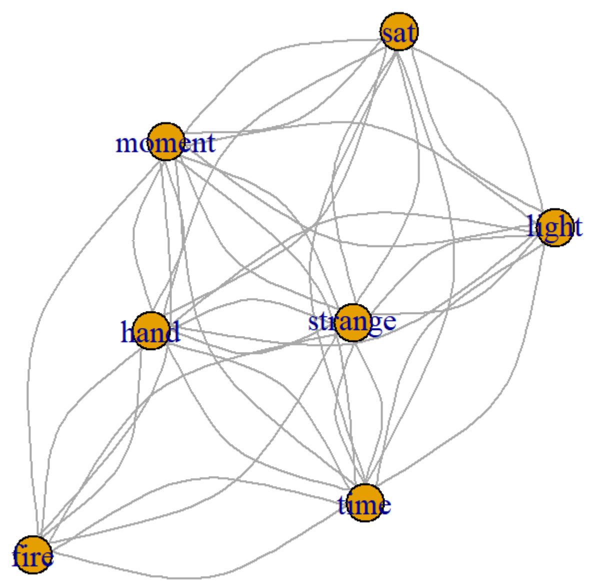
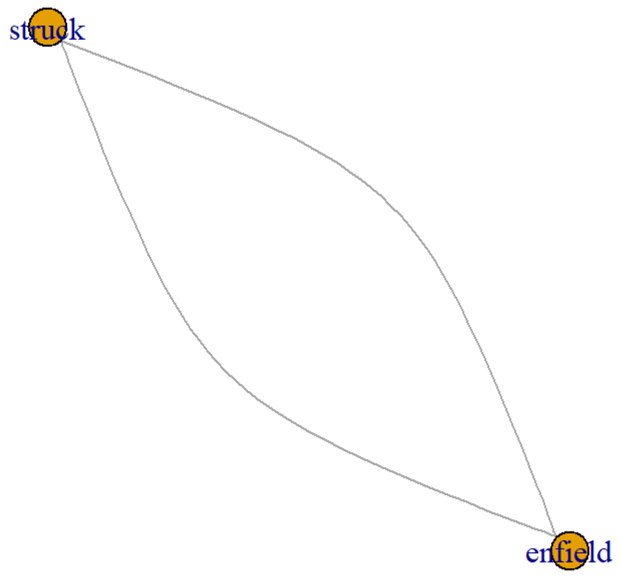


Fig 15.10edge betweeness based community 10
sat
moment
light
hand
strange
time
fire

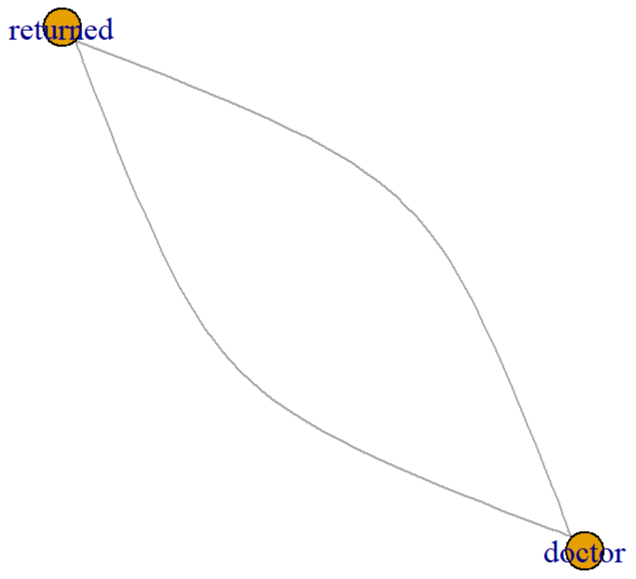
## Fig 15.11edge betweeness based community 11



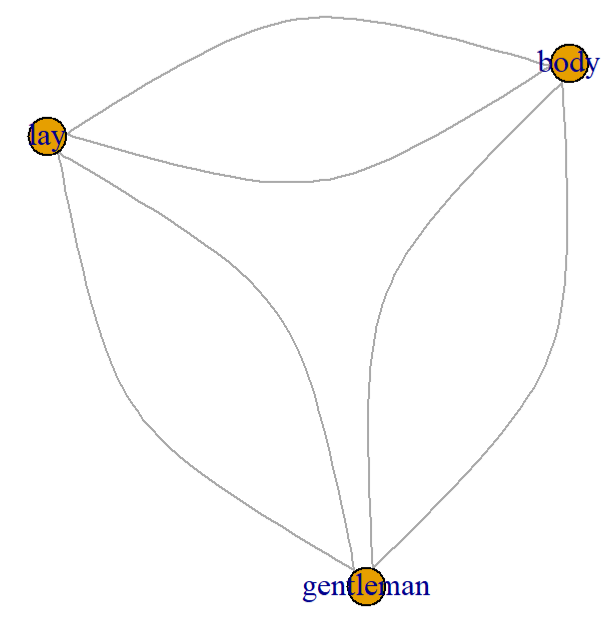
## Fig 15.12edge betweeness based community 12

## Fig 15.13edge betweeness based community 13

## Fig 15.14edge betweeness based community 14

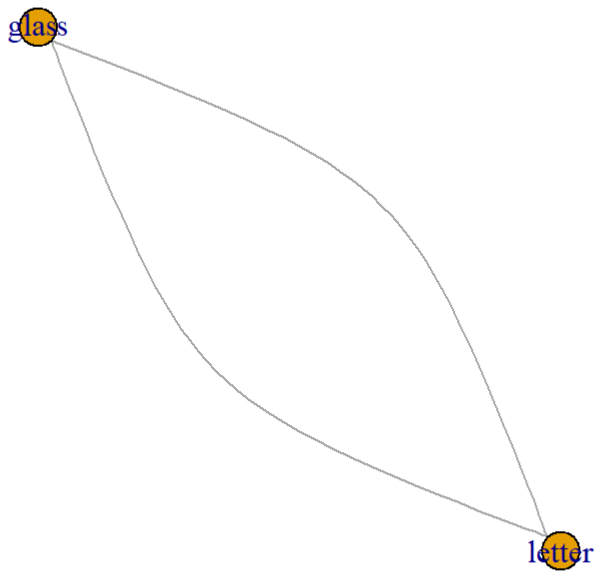


## Fig 15.15edge betweeness based community 15



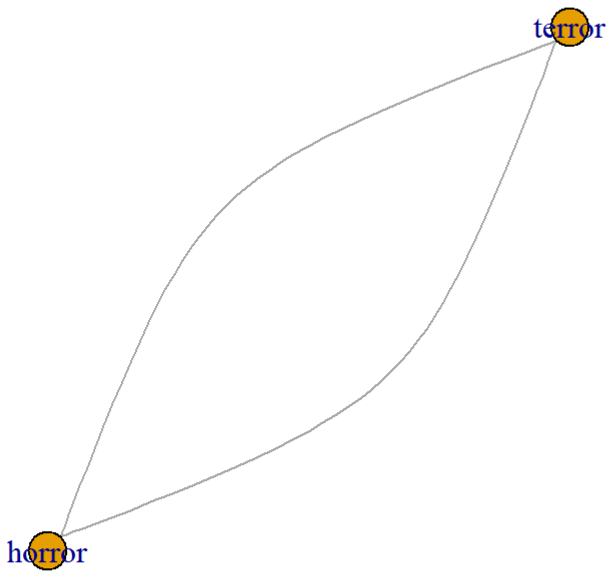
## Fig 15.16edge betweeness based community 16

## Fig 15.17edge betweeness based community 17

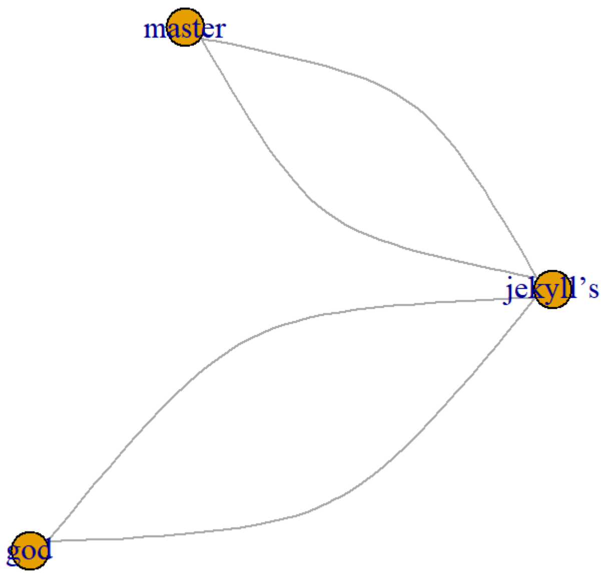


## Fig 15.18edge betweeness based community 18

## Fig 15.19edge betweeness based community 19



## Fig 15.20edge betweeness based community 20



## Fig 15.21edge betweeness based community 21

## Fig 15.22edge betweeness based community 22

## Fig 15.23edge betweeness based community 23

## Fig 15.24edge betweeness based community 24

## Fig 15.25edge betweeness based community 25

## Fig 15.26edge betweeness based community 26

The individual community plots are obtained and seen to understand the approach.

It is finally decided to go ahead with Louvian algorithm. This is because it is having proper and meaningful sub graphs and also the highest modularity score among all others present.

# 5. Natural Language Processing of the book

Based on NLP, we can understand the context of the words in a much better manner. Inorder to do NLP tasks, two approaches namely sentiment analysis and ngram are considered. This is done by the help of the tidytext package and the basic examples given in its official package site.

https://www.tidytextmining.com/sentiment.html (https://www.tidytextmining.com/sentiment.html)

https://www.tidytextmining.com/ngrams.html (https://www.tidytextmining.com/ngrams.html)

## 5.1 Sentiment Analysis

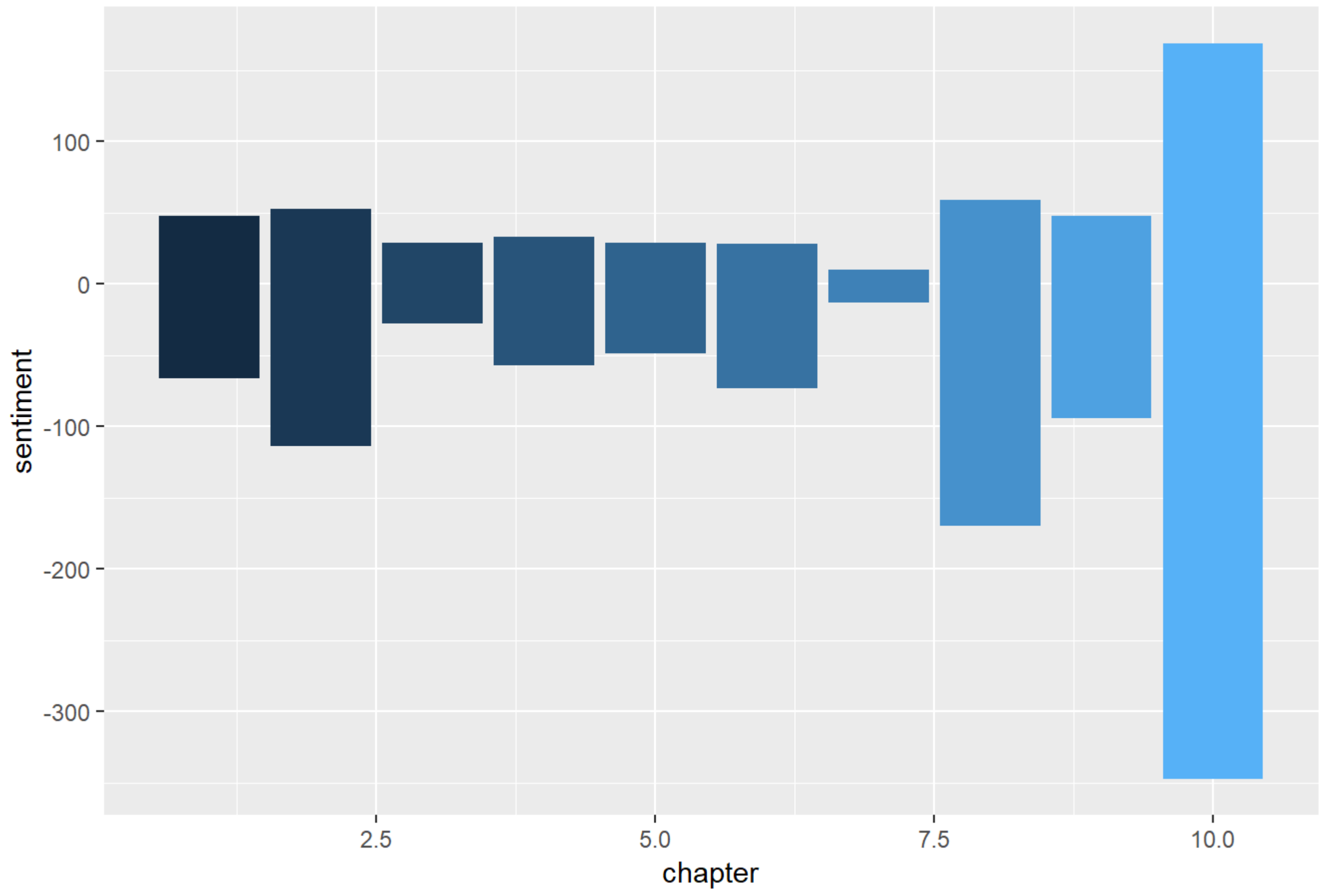
In order to perform the sentiment analysis, there are 3 main approachs present namely bing,afinn and nrc. Among them, bing and nrc approaches are to be considered for this book. It is done by considering unigram and therefore only single word per occurence is considered.

Now, based on bing, the sentiment of the plot through is found out per each chapter. In this, words are categorised based on whether they are positive or negative on the context and are given pre defined weights. Now, the words present in the book is inner joined with the sentiment weights that are defined already. This helps in establishing the overall mood of the book through the chapters. The sentiment is calculated as the difference between positive and negative and the resulting plot is plotted.

```
sentiment1<- book %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, chapter,sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

```
ggplot(sentiment1, aes(chapter, sentiment, fill = chapter)) +
  geom_col(show.legend = FALSE)+
  ggtitle("Fig 16 Sentiment analysis based on bing")
```

Fig 16 Sentiment analysis based on bing



From this it is seen that the overall mood of the story in the book is negative which gradually grows into negative sentiment that peaks in the climax.

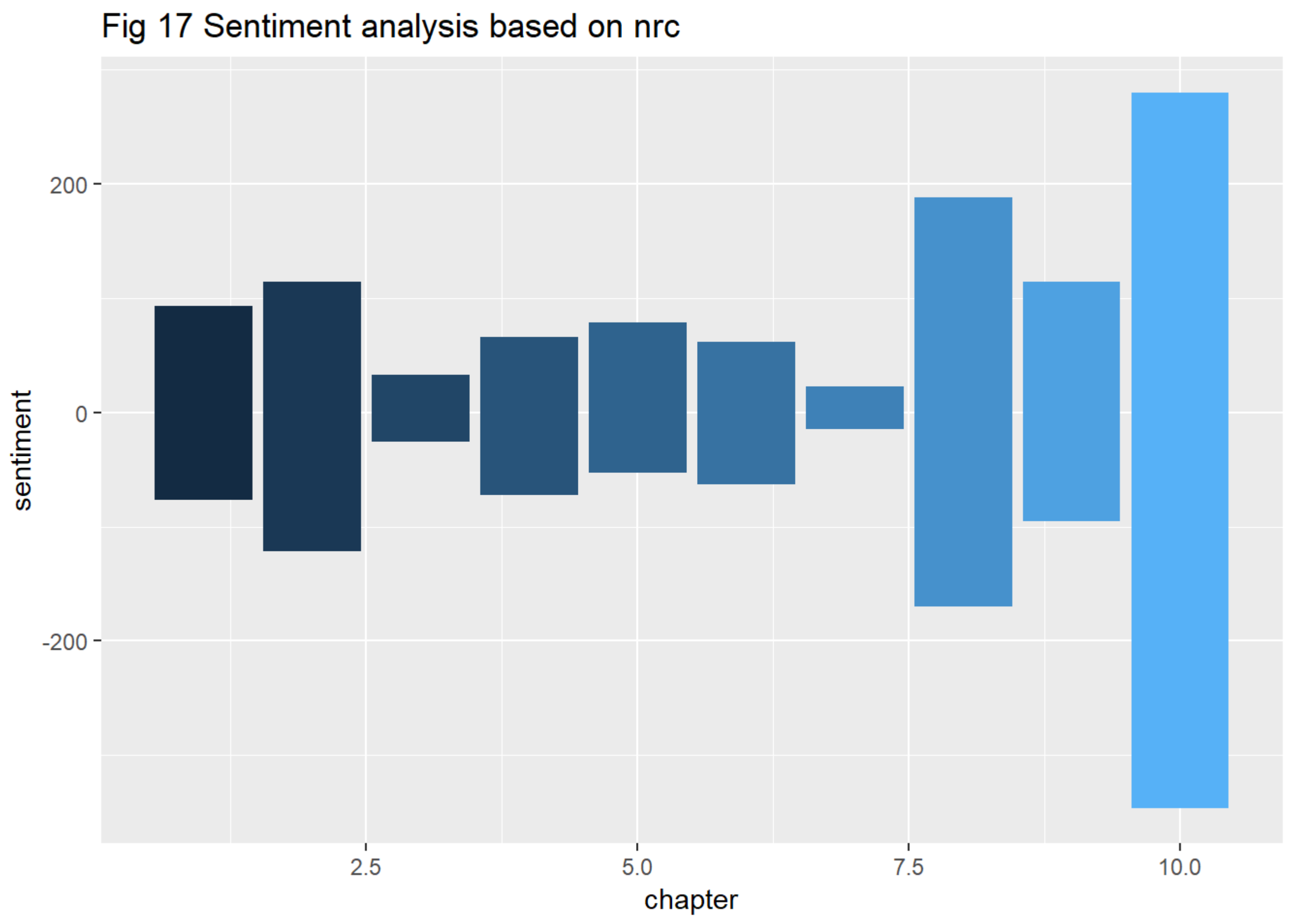
The same approach is repeated with the nrc sentiment weights and checked. In this, the weights are given negative if it expresses any one of bad traits like anger,sadness etc and positive if it expresses any one of the good traits like happy,joyful etc.

```
head(sentiment1)
```

```
## # A tibble: 6 x 5
##   word           chapter negative positive sentiment
##   <chr>            <int>    <dbl>    <dbl>     <dbl>
## 1 abnormal             9        1        0        -1
## 2 abominable           3        1        0        -1
## 3 abruptly             9        1        0        -1
## 4 absence              2        1        0        -1
## 5 accomplishment       4        0        1         1
## 6 accursed             6        1        0        -1
```

```
sentiment2<- book %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, chapter,sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

```
ggplot(sentiment2, aes(chapter, sentiment, fill = chapter)) +
  geom_col(show.legend = FALSE)+
  ggtitle("Fig 17 Sentiment analysis based on nrc")
```



It is seen that the sentiment flowthrough is similar to bing but in this the lines are seen to be distributed due to the different approach.But it reconfirms that the overall mood of the book is negative.

Now, based on this, we can also segregate the words as positive and negative based on the sentiment they exhibit. This can be done by filtering them based on their sentiment projected.

```
positive <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

  book %>%
  inner_join(positive) %>%
  count(word, sort = TRUE)
```

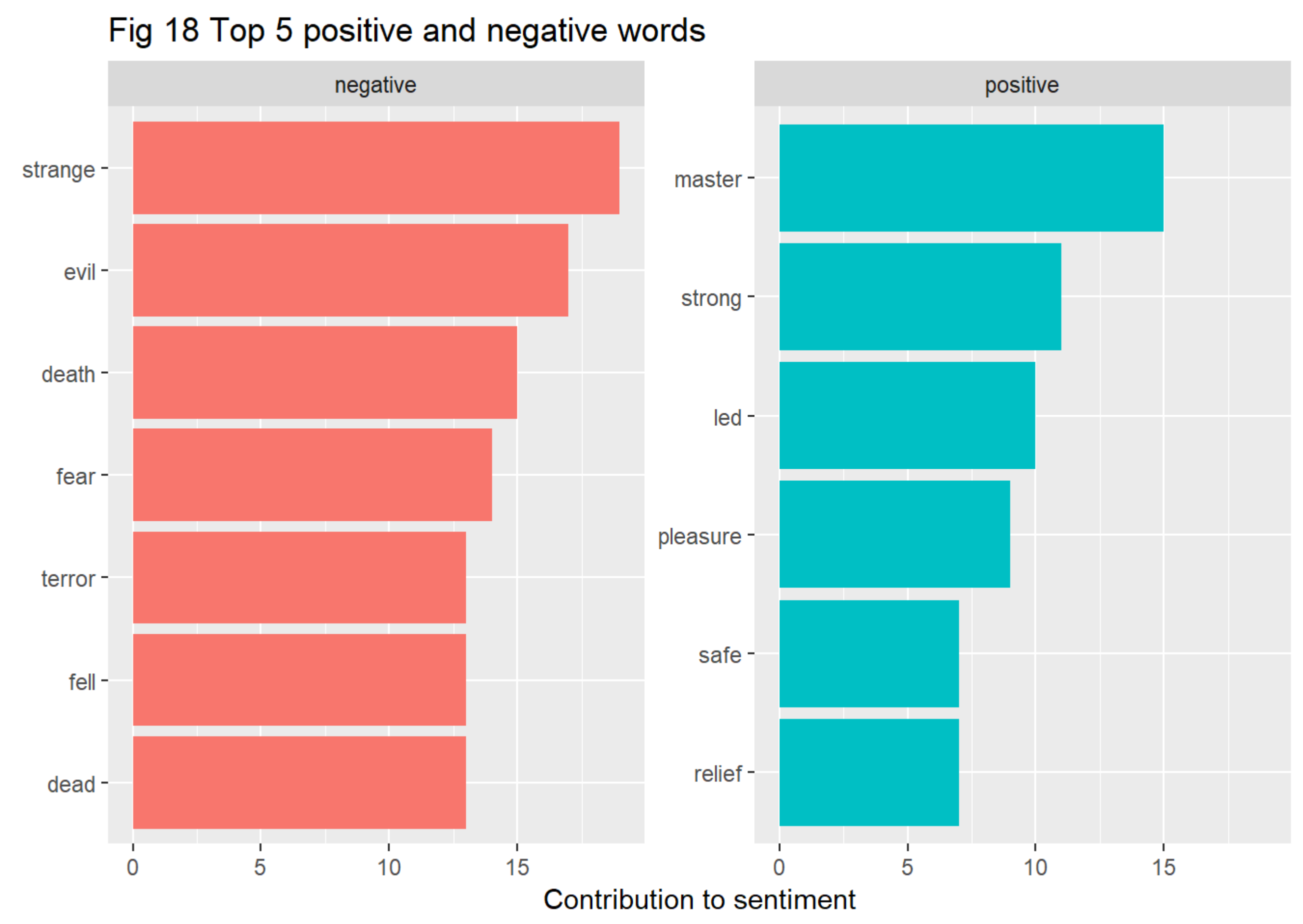
```
## # A tibble: 136 x 2
##    word          n
##    <chr>     <int>
##  1 god          22
##  2 friend       20
##  3 found        17
##  4 child         9
##  5 visitor       9
##  6 white         8
##  7 hope          7
##  8 safe          7
##  9 deal          6
## 10 excellent     6
## # ... with 126 more rows
```

The word counts of the word and their overall sentiment are obtained, This helps in understanding that what negative words are used frequently and what positive words are used frequently and what are their occurences in total.

```
wordcount <- book %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

wordcount
```

```
## # A tibble: 778 x 3
##    word    sentiment     n
##    <chr>   <chr>     <int>
##  1 strange negative     19
##  2 evil    negative     17
##  3 death   negative     15
##  4 master  positive     15
##  5 fear    negative     14
##  6 dead    negative     13
##  7 fell    negative     13
##  8 terror  negative     13
##  9 struck  negative     12
## 10 strong  positive     11
## # ... with 768 more rows
```

```
sentiplot<-wordcount %>%
  group_by(sentiment) %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  ggtitle("Fig 18 Top 5 positive and negative words")+
  coord_flip()
sentiplot
```



From this, it is seen that most occuring negative words are strange,evil,fear and death which denotes again that the story of the book deals with a negative aspect and there is several things involving death and its subsequent effects.

## 5.2 ngram

The ngram is done by modifying the unnest_tokens() function given at the top for the correlation graph. When the token is given as ngram inside it, ngram function is activated and when n is two, it is bigram and when n is 3 it is trigram. It is then separated to 2 or 3 words as per the ngram being performed and the stop words are filtered out and after which it is united together and rbinded.

## bigram

The bigram is performed by considering n to be as two and following the other steps as told above.

```
book_title = "thestrangecaseofdrjekyllandmrhyde"
chapters = 1:10
chapter_stem = "thestrangecaseofdrjekyllandmrhyde"
ext <-".txt"
folder<- "./thestrangecaseofdrjekyllandmrhyde/"

bigram <- tibble()
```

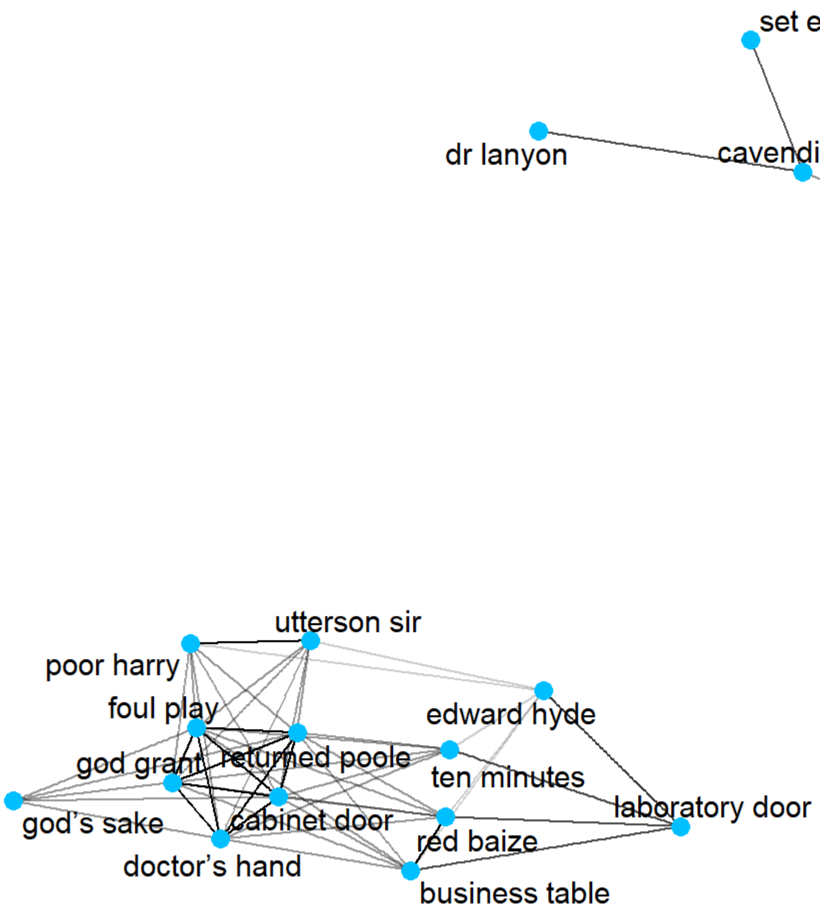
```
for(i in chapters){

  chapter <-paste0(folder,chapter_stem,i,ext)

  raw<-readChar(chapter, file.info(chapter)$size)

  chapter_text <- raw %>%
    gsub("[\r\n]+", " ", .) %>%
    gsub('^"',"",.) %>%
    gsub('"$',"",.)

  #creates a tibble with 3 cols: book_title, chapter, word
  words <- tibble(title = book_title, chapter=i, text = chapter_text) %>%
    unnest_tokens(bigram, text, token = "ngrams", n = 2) %>% # tokenise the text
    separate(bigram, c("word1", "word2"), sep = " ") %>%
    filter(!word1 %in% stop_words$word,
           !word2 %in% stop_words$word) %>% # remove stop words
    unite("bigram", c("word1", "word2"), sep = " ")

  bigram <- rbind(bigram, words) # add rows to the book tibble
}
```

The bigram correlation is then found out for the bigram words with the condition that the number of word pairs minimum is 3 and the correlations is above 0.6. The resulting correlation graph is then plotted as using both text and label .

```
bigram_cor <- bigram %>%
   group_by(bigram) %>%
   filter(n() >= 3) %>% # minimum number of word pairs to consider; determines the number of nod
es; lower value -> more nodes
   pairwise_cor(item=bigram, feature=chapter) %>%
   filter(!is.na(correlation), correlation >= 0.60)
```

```
bigram_cor_graph<-graph_from_data_frame(bigram_cor)


  ggraph(bigram_cor_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_text(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alpha
= 0.7, segment.size = 0.4) +
    labs(title="Fig 19 bigram for the book")+# repel =TRUE stops labels overlapping; but also  a
ttaches a segment line between node and text, which may be visually confusing if it overlaps wit
h graph edges and nodes
  theme_void()
```

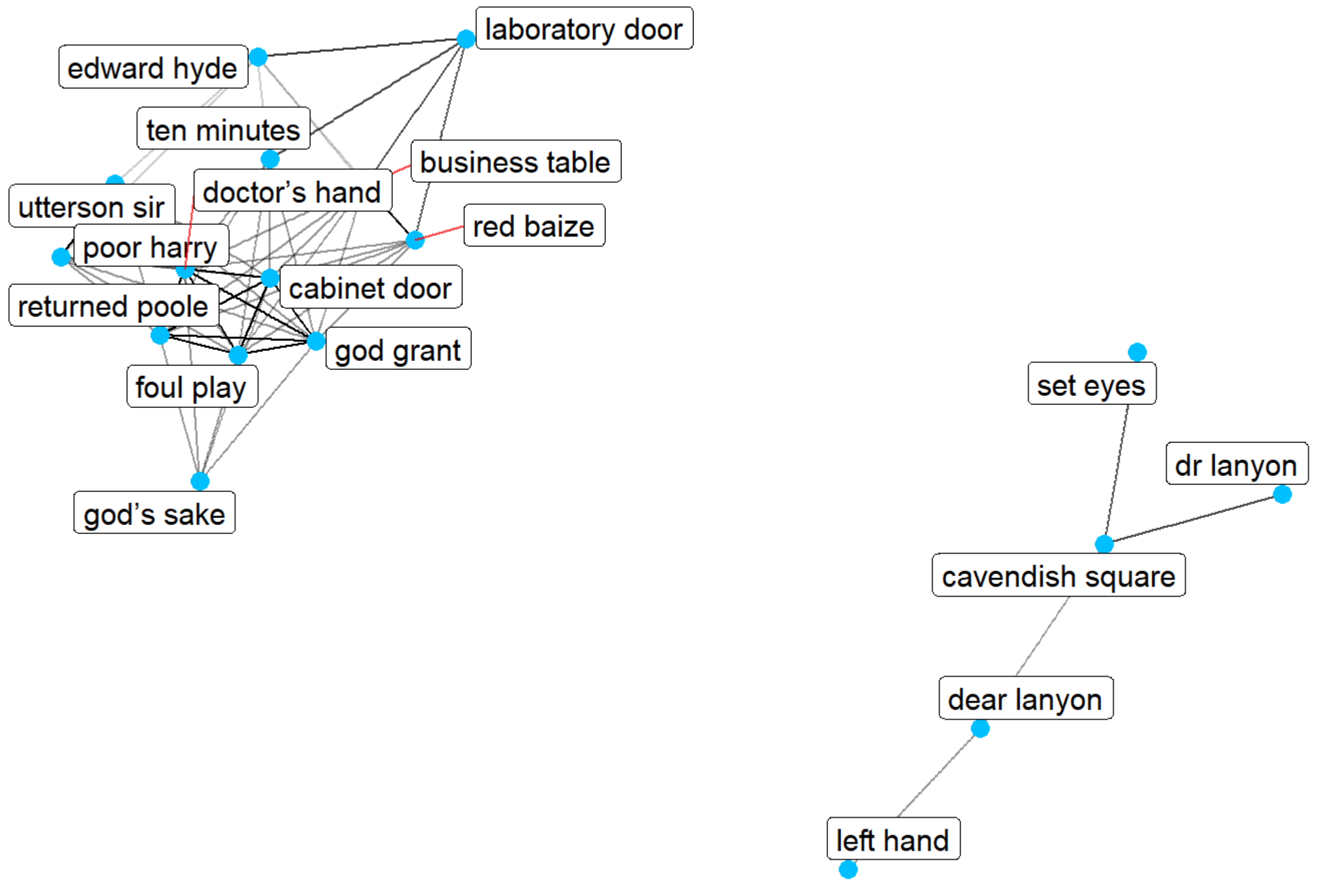
Fig 19 bigram for the book



From this it is seen that there is one strongly connected cluster and the other one to be present with lesser terms. From this connectors, the interaction between the main characters and the events are seen clearly.

```
ggraph(bigram_cor_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_label(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alph
a = 0.7, segment.size = 0.4) +
  labs(title="Fig 20 bigram for the book")+# repel =TRUE stops labels overlapping; but also  att
aches a segment line between node and text, which may be visually confusing if it overlaps with
 graph edges and nodes
  theme_void()
```

Fig 20 bigram for the book



The diameter,radius ,degree and eccentricity of the bigram correlation graph is calculated. From this it is seen that bigrams with large degree are present together in the cluster.

```
print('The diameter of the graph is')
```

```
## [1] "The diameter of the graph is"
```

```
print(diameter(bigram_cor_graph))
```

```
## [1] 3
```

```
print("The radius of the graph is")
```

```
## [1] "The radius of the graph is"
```

```
print(radius(bigram_cor_graph))
```

```
## [1] 2
```

```
print("The degrees are")
```

```
## [1] "The degrees are"
```

```
print(degree(bigram_cor_graph))
```

```
##      dear lanyon     utterson sir       poor harry        red baize 
##                4               14               14               16 
##   business table  laboratory door      ten minutes        dr lanyon 
##               16                8               14                2 
##         set eyes cavendish square      edward hyde   returned poole 
##                2                6               12               20 
##        foul play        god grant     cabinet door    doctor’s hand 
##               20               20               20               20 
##       god’s sake        left hand 
##               10                2
```

```
print("The eccentricity is")
```

```
## [1] "The eccentricity is"
```

```
print(eccentricity(bigram_cor_graph))
```

```
##      dear lanyon     utterson sir       poor harry        red baize 
##                2                2                2                2 
##   business table  laboratory door      ten minutes        dr lanyon 
##                2                3                2                3 
##         set eyes cavendish square      edward hyde   returned poole 
##                3                2                3                2 
##        foul play        god grant     cabinet door    doctor’s hand 
##                2                2                2                2 
##       god’s sake        left hand 
##                3                3
```

## trigram

For trigram, n is considered to be 3 and rest of the same steps as bigram is followed

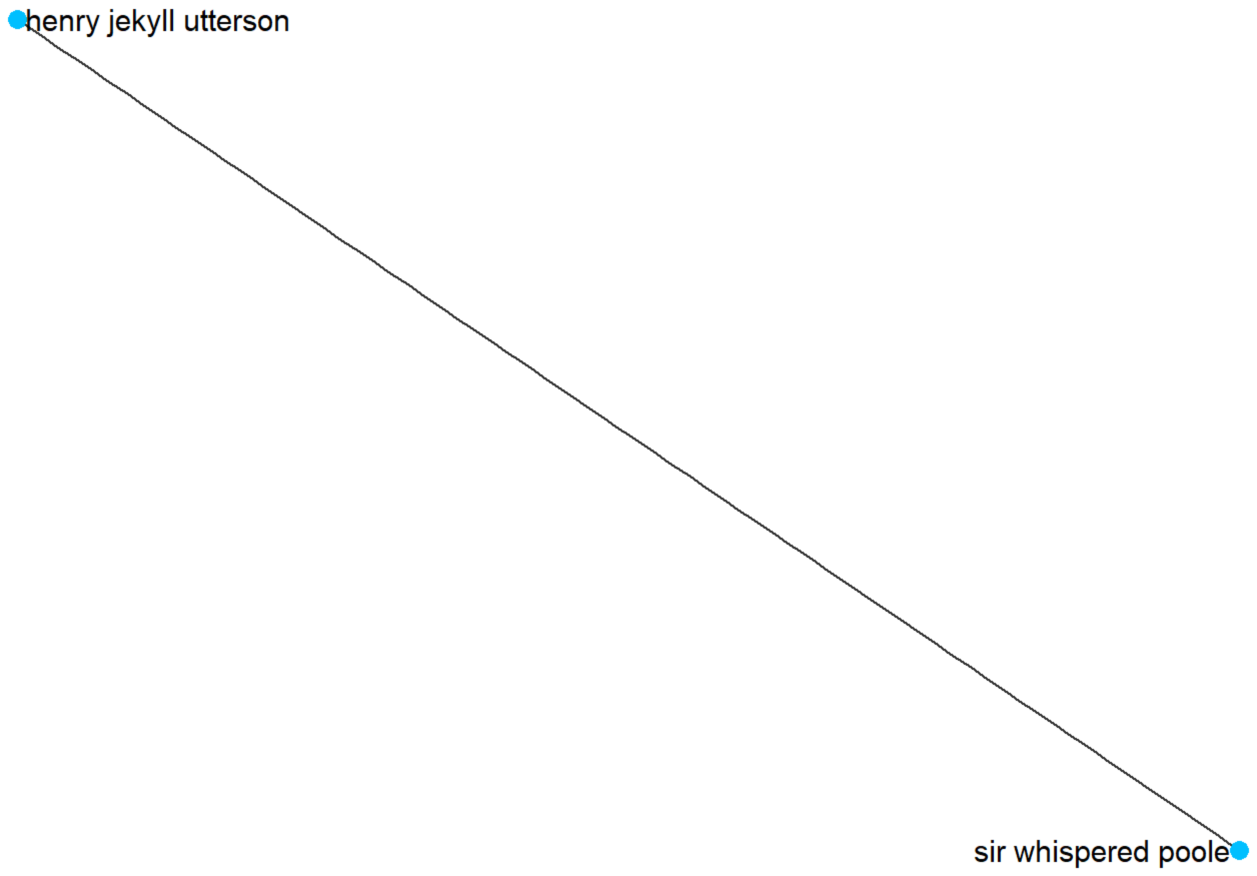
```
book_title = "thestrangecaseofdrjekyllandmrhyde"
chapters = 1:10
chapter_stem = "thestrangecaseofdrjekyllandmrhyde"
ext <-".txt"
folder<- "./thestrangecaseofdrjekyllandmrhyde/"

trigram <- tibble()
```

```
for(i in chapters){

  chapter <-paste0(folder,chapter_stem,i,ext)

  raw<-readChar(chapter, file.info(chapter)$size)

  chapter_text <- raw %>%
    gsub("[\r\n]+", " ", .) %>%
    gsub('^"',"",.) %>%
    gsub('"$',"",.)

  #creates a tibble with 3 cols: book_title, chapter, word
  words <- tibble(title = book_title, chapter=i, text = chapter_text) %>%
    unnest_tokens(trigram, text, token = "ngrams", n = 3) %>% # tokenise the text
    separate(trigram, c("word1", "word2","word3"), sep = " ") %>%
    filter(!word1 %in% stop_words$word,
           !word2 %in% stop_words$word,
           !word3 %in% stop_words$word) %>% # remove stop words
    unite("trigram", c("word1", "word2","word3"), sep = " ")

  trigram <- rbind(trigram, words) # add rows to the book tibble
}
```

```
trigram_cor <- trigram %>%
  group_by(trigram) %>%
  filter(n() >= 2) %>% # minimum number of word pairs to consider; determines the number of node
s; lower value -> more nodes
  pairwise_cor(item=trigram, feature=chapter) %>%
  filter(!is.na(correlation), correlation >= 0.50)
```

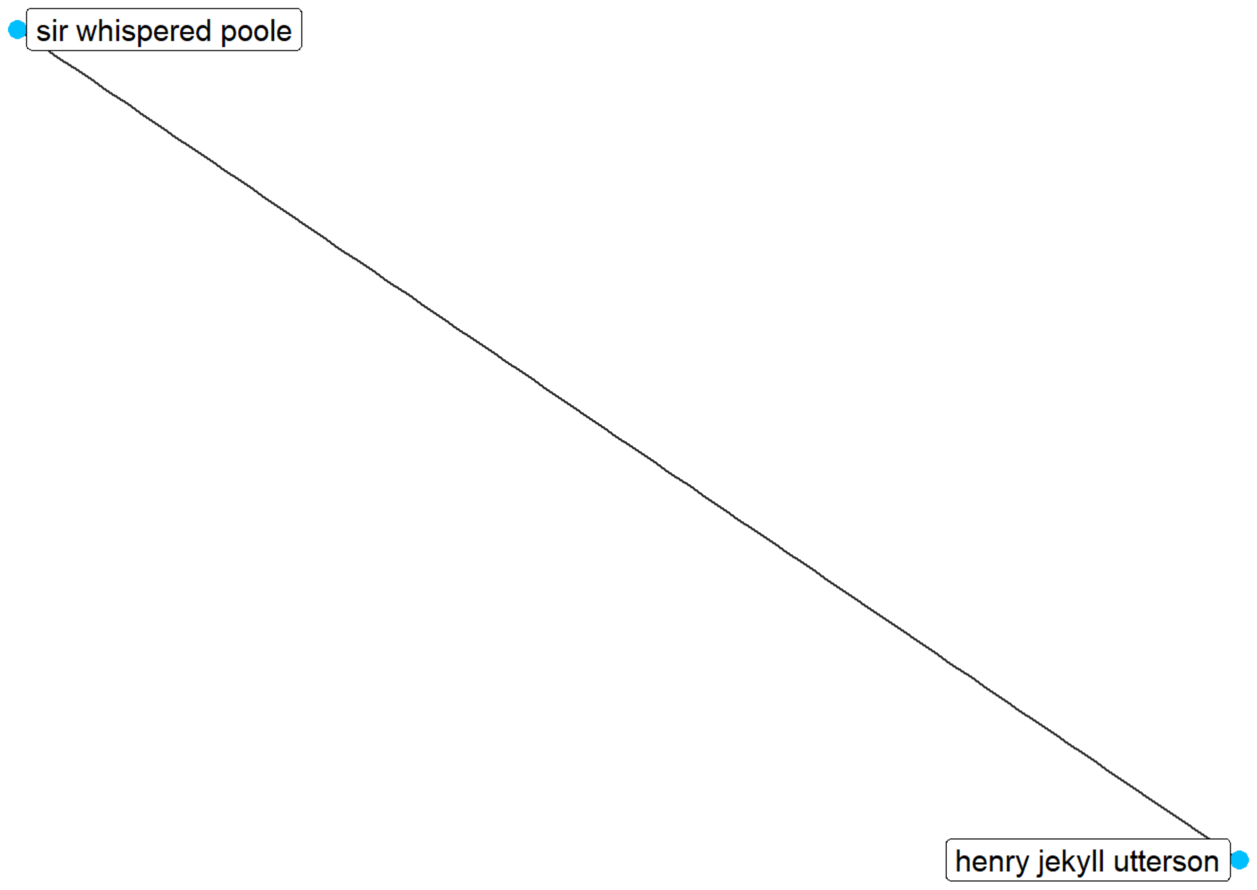
```
trigram_cor_graph<-graph_from_data_frame(trigram_cor)



  ggraph(trigram_cor_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_text(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alpha
= 0.7, segment.size = 0.4) +
    labs(title="Fig 21 trigram for the book")+# repel =TRUE stops labels overlapping; but also
 attaches a segment line between node and text, which may be visually confusing if it overlaps w
ith graph edges and nodes
  theme_void()
```

Fig 21 trigram for the book



```
ggraph(trigram_cor_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_label(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alph
a = 0.7, segment.size = 0.4) +
  labs(title="Fig 22 trigram for the book")+# repel =TRUE stops labels overlapping; but also  at
taches a segment line between node and text, which may be visually confusing if it overlaps with
graph edges and nodes
  theme_void()
```

## Fig 22 trigram for the book



Here it is seen that even when the minimum pair is given as 2 and correlation as 0.5, only two trigrams are satisfying the condition. From this, no meaningful information can be found out.

THe diameter,radius,degree and eccentricity are found. It is seen there provide no purpose for understanding the trigram.

```
print('The diameter of the graph is')
```

```
## [1] "The diameter of the graph is"
```

```
print(diameter(trigram_cor_graph))
```

```
## [1] 1
```

```
print("The radius of the graph is")
```

```
## [1] "The radius of the graph is"
```

```
print(radius(trigram_cor_graph))
```

```
## [1] 1
```

```
print("The degrees are")
```

```
## [1] "The degrees are"
```

```
print(degree(trigram_cor_graph))
```

```
##   sir whispered poole henry jekyll utterson
##                     2                     2
```

```
print("The eccentricity is")
```

```
## [1] "The eccentricity is"
```

```
print(eccentricity(trigram_cor_graph))
```

```
##   sir whispered poole henry jekyll utterson
##                     1                     1
```

## 6.Comparison of Book by section

In this part, the book is divided into two sections namely first half with the first 5 chapter and second half with the last 5 chapters. This is done inorder to see the principle relationships extracted and compare them between the two halves.

### 6.1 First half of the book

Now, the chapters of 1 to 5 are considered for this. All the steps of followed for the book in correlation graph is repeated here again. the louvian algorithm alone is used for community detection as it is found to be the efficient for the given book and the corresponding graph measures are calculated.

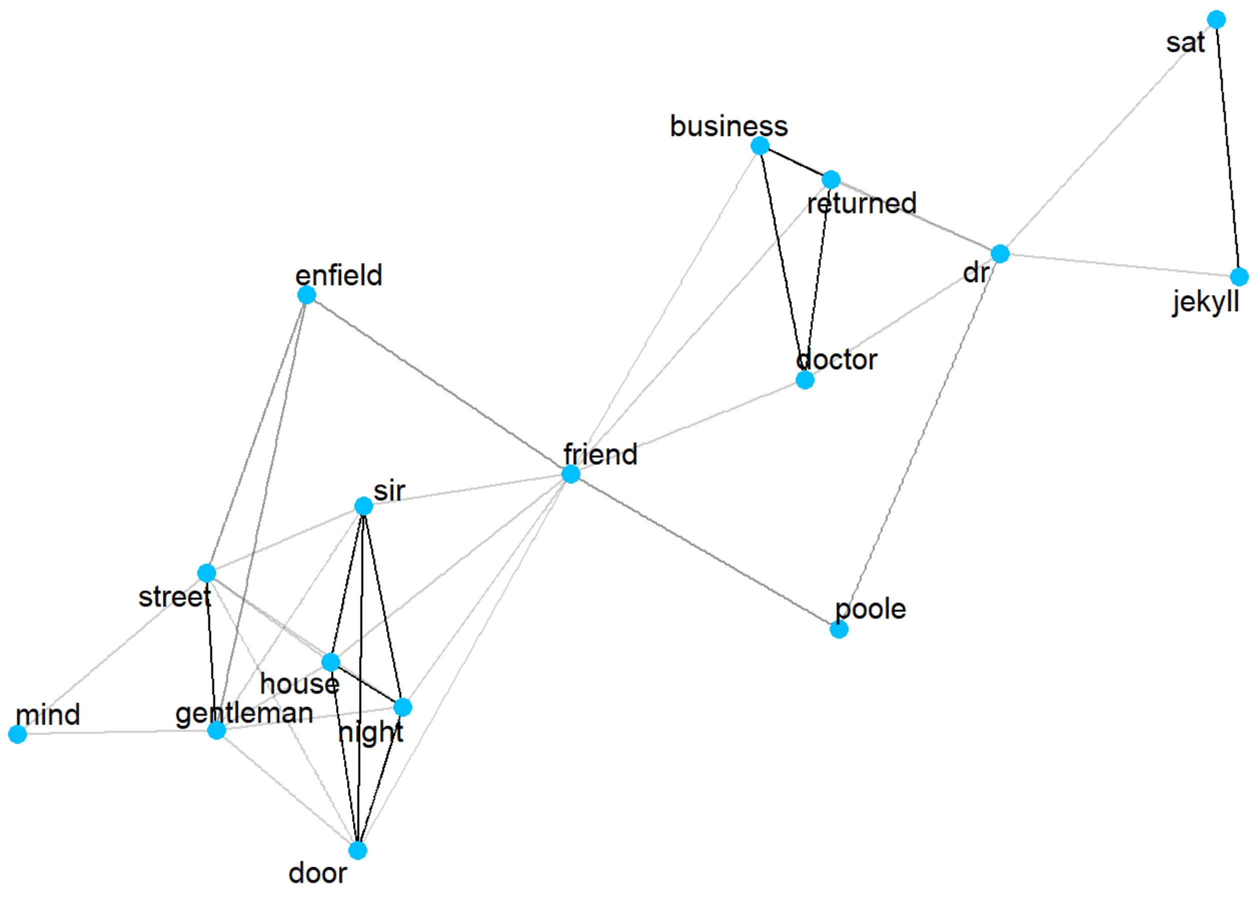
```
book_title = "thestrangecaseofdrjekyllandmrhyde"
chapters = 1:5
chapter_stem = "thestrangecaseofdrjekyllandmrhyde"
ext <-".txt"
folder<- "./thestrangecaseofdrjekyllandmrhyde/"

book <- tibble()
```

```
#read in each chapter
for(i in chapters){

  chapter <-paste0(folder,chapter_stem,i,ext)

  raw<-readChar(chapter, file.info(chapter)$size)

  chapter_text <- raw %>%
    gsub("[\r\n]+", " ", .) %>%
    gsub('^"',"",.) %>%
    gsub('"$',"",.)

  #creates a tibble with 3 cols: book_title, chapter, word
  words <- tibble(title = book_title, chapter=i, text = chapter_text) %>%
    unnest_tokens(word, text) %>% # tokenise the text
    filter(!word %in% stop_words$word) # remove stop words

  book <- rbind(book, words) # add rows to the book tibble

}
```

```
word_cor <- book %>%
  group_by(word) %>%
  filter(n() >= 10) %>% # minimum number of word pairs to consider; determines the number of nod
es; lower value -> more nodes
  pairwise_cor(item=word, feature=chapter) %>%
  filter(!is.na(correlation), correlation >= 0.60) # minimum correlation; determines the number
 of edges

correlation_graph <-graph_from_data_frame(word_cor,directed = FALSE)
lyt <- layout_with_kk(correlation_graph)
```

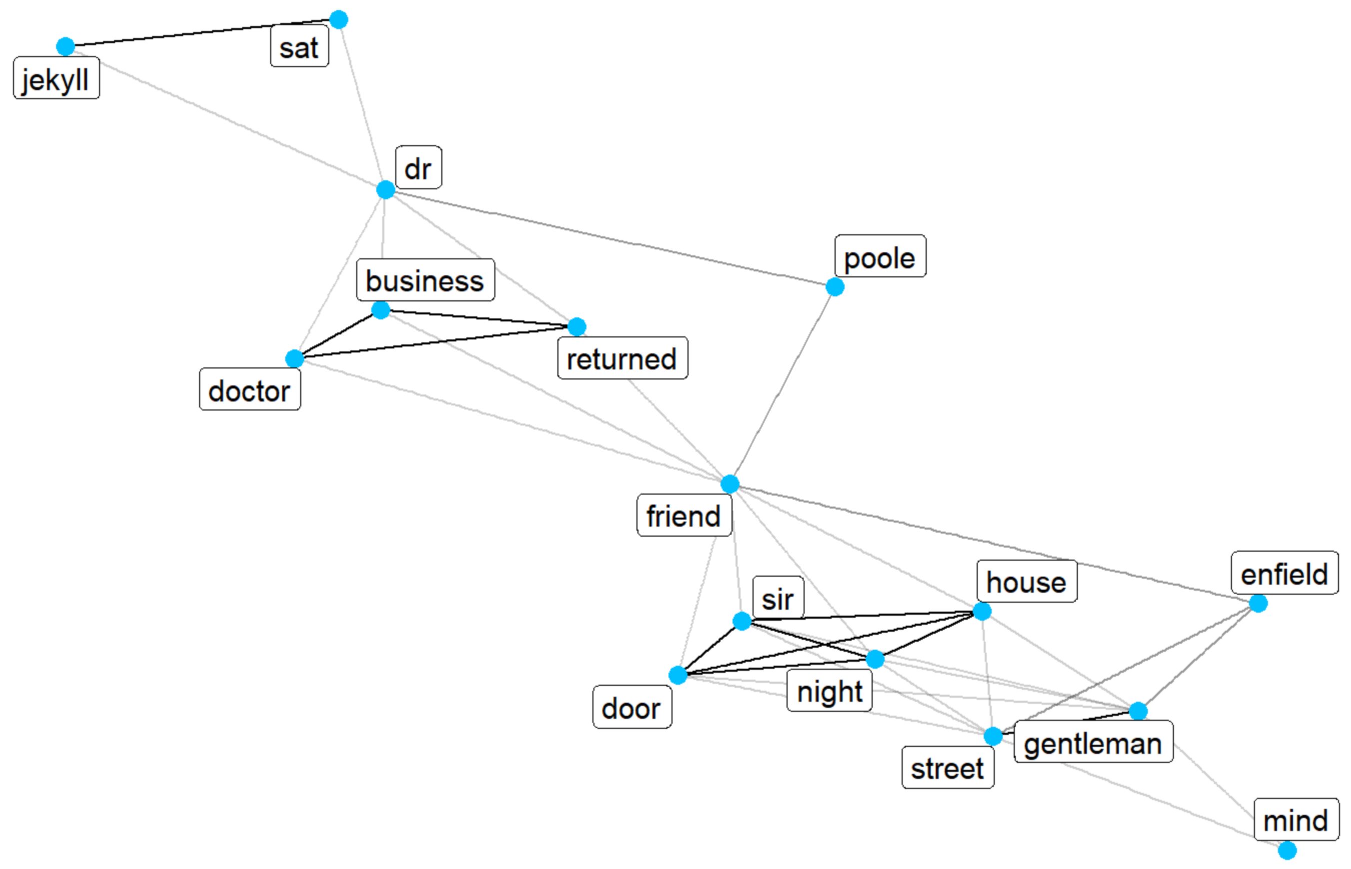
```
ggraph(correlation_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_text(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alpha
= 0.7, segment.size = 0.4) +
  labs(title="Fig 23 Correlatin graph for the first half of the book")+# repel =TRUE stops label
s overlapping; but also  attaches a segment line between node and text, which may be visually co
nfusing if it overlaps with graph edges and nodes
  theme_void()
```

Fig 23 Correlatin graph for the first half of the book



```
ggraph(correlation_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_label(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alph
a = 0.7, segment.size = 0.4) +
  labs(title="Fig 24 Correlatin graph for the first half of the book")+# repel =TRUE stops label
s overlapping; but also  attaches a segment line between node and text, which may be visually co
nfusing if it overlaps with graph edges and nodes
  theme_void()
```

Fig 24 Correlatin graph for the first half of the book



```
print('The diameter of the graph is')
```

```
## [1] "The diameter of the graph is"
```

```
print(diameter(correlation_graph))
```

```
## [1] 6
```

```
print("The radius of the graph is")
```

```
## [1] "The radius of the graph is"
```

```
print(radius(correlation_graph))
```

```
## [1] 3
```

```
print("The degrees are")
```

```
## [1] "The degrees are"
```

```
print(degree(correlation_graph))
```

```
##    friend    street gentleman   enfield  business      door  returned 
##        18        14        14         6         8        12         8 
##       sir    doctor     night     house     poole        dr      mind 
##        12         8        12        12         4        12         4 
##    jekyll       sat 
##         4         4
```

```
print("The eccentricity is")
```

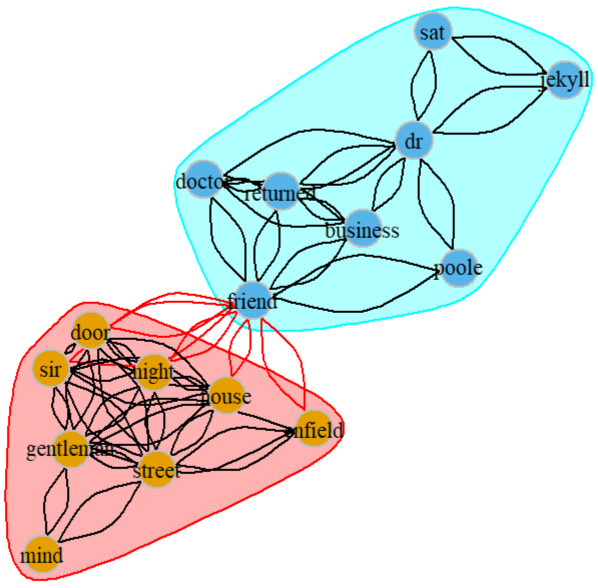
```
## [1] "The eccentricity is"
```

```
print(eccentricity(correlation_graph))
```

```
##    friend    street gentleman   enfield  business      door  returned 
##         3         5         5         4         4         4         4 
##       sir    doctor     night     house     poole        dr      mind 
##         4         4         4         4         4         5         6 
##    jekyll       sat 
##         6         6
```

```
example.louv<- cluster_louvain(correlation_graph)
# membership vector
b<-membership(example.louv)


mod_max <- example.louv$modularity
k_opt <- length(example.louv)

# we can  use the plot.igraph approach to visualise the communities
plot(example.louv, correlation_graph, layout=lyt ,vertex.frame.color="gray", vertex.label.color=
"black", vertex.label.cex=0.7,main = paste("Fig 25 Louvain k=", max(k_opt), "; Modularity=", max
(round(mod_max,2))))
```

**Fig 25 Louvain k= 2 ; Modularity= 0.36**



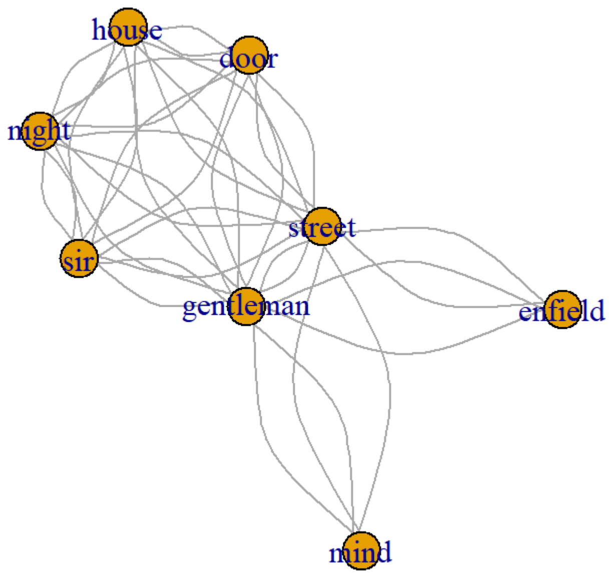
```
source("my_modularity.R")
source("cluster_measurements.R")
intra.cluster.df<-cluster_measurements(correlation_graph,example.louv)
knitr::kable(intra.cluster.df, align = 'l', caption = paste0(algorithm(example.louv),  " : size,
modularity and density scores of communities detected"))
```

multi level : size, modularity and density scores of communities detected

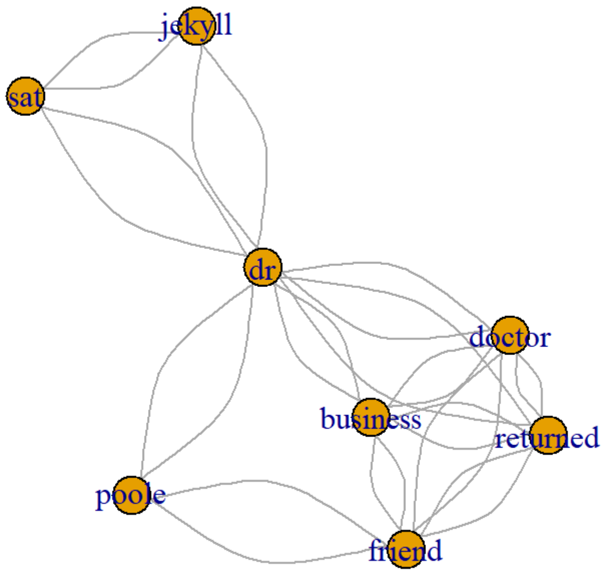
| cluster | size | modularity | density |
|---|---|---|---|
| -1 | 16 | 0.000000 | 0.6333333 |
| 1 | 8 | 0.250000 | 1.3571429 |
| 2 | 8 | 0.232687 | 1.0000000 |

```
par(mfrow=c(1,1))
for(i in 1:length(unique(example.louv$membership))){
  plot(induced.subgraph(correlation_graph,example.louv$names[example.louv$membership==i]))
  title(paste0("Fig 26.",i,"louvian based community ",i))
}
```

## Fig 26.1louvian based community 1



## Fig 26.2louvian based community 2



Totally 2 communities are obtained.

The first community tells about the incident happening at night time in a house.

The second community is about dr.jekyll and describes about his profession and friends.

It is seen that the flow throught here is found to be similar to the initial part of the louvian algorithm done to the book and the communities here are one third of the total communities in the book.

## 6.2 Second half of the book

The same processes as above are repeated for the chapters 5 to 10 as the second half of the book

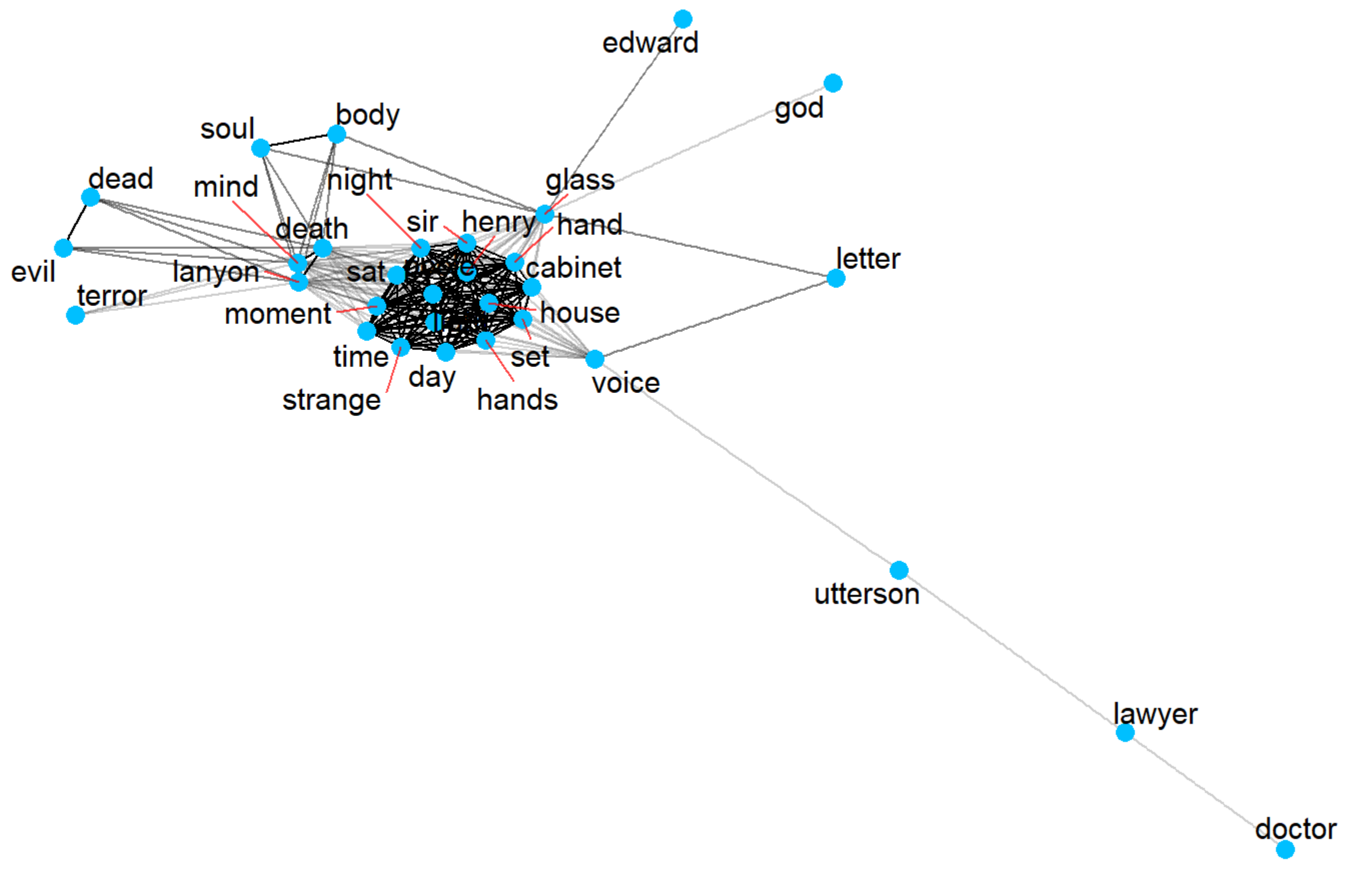
```
book_title = "thestrangecaseofdrjekyllandmrhyde"
chapters = 5:10
chapter_stem = "thestrangecaseofdrjekyllandmrhyde"
ext <-".txt"
folder<- "./thestrangecaseofdrjekyllandmrhyde/"

book <- tibble()
```

```
#read in each chapter
for(i in chapters){

  chapter <-paste0(folder,chapter_stem,i,ext)

  raw<-readChar(chapter, file.info(chapter)$size)

  chapter_text <- raw %>%
    gsub("[\r\n]+", " ", .) %>%
    gsub('^"',"",.) %>%
    gsub('"$',"",.)

  #creates a tibble with 3 cols: book_title, chapter, word
  words <- tibble(title = book_title, chapter=i, text = chapter_text) %>%
    unnest_tokens(word, text) %>% # tokenise the text
    filter(!word %in% stop_words$word) # remove stop words

  book <- rbind(book, words) # add rows to the book tibble

}
```

```
word_cor <- book %>%
  group_by(word) %>%
  filter(n() >= 12) %>% # minimum number of word pairs to consider; determines the number of nod
es; lower value -> more nodes
  pairwise_cor(item=word, feature=chapter) %>%
  filter(!is.na(correlation), correlation >= 0.60) # minimum correlation; determines the number
 of edges

correlation_graph <-graph_from_data_frame(word_cor,directed = FALSE)
lyt <- layout_with_kk(correlation_graph)
```

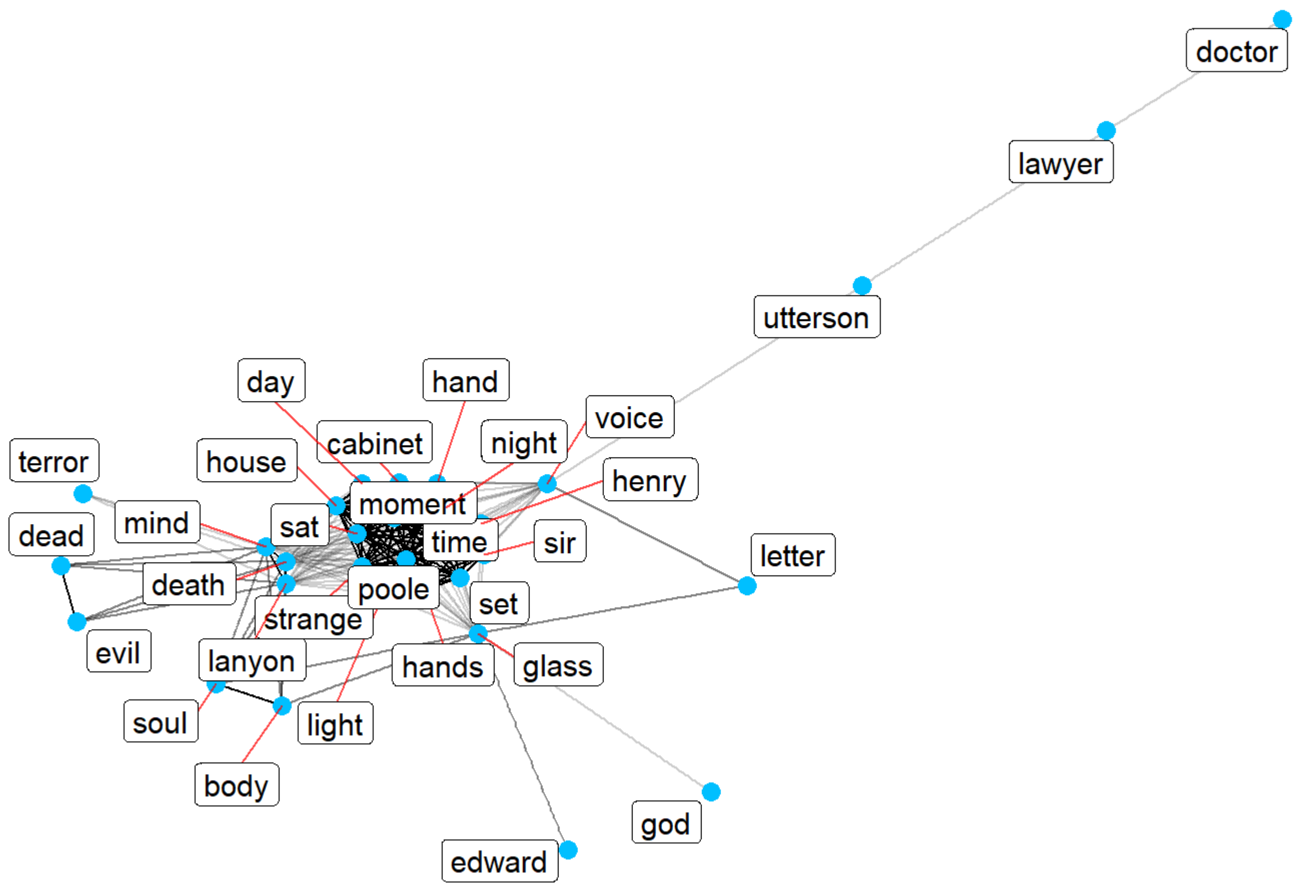
```
ggraph(correlation_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_text(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alpha
= 0.7, segment.size = 0.4) +
  labs(title="Fig 27 Correlatin graph for the second half of the book")+# repel =TRUE stops labe
ls overlapping; but also  attaches a segment line between node and text, which may be visually c
onfusing if it overlaps with graph edges and nodes
  theme_void()
```

Fig 27 Correlatin graph for the second half of the book



```
ggraph(correlation_graph, layout = "fr") +
  geom_edge_link2(aes(edge_alpha = correlation), edge_width=0.5, show.legend = FALSE) +
  geom_node_point(color = "deepskyblue", size = 3) +
  geom_node_label(aes(label = name), size = 4, repel = TRUE, segment.color = "red", segment.alph
a = 0.7, segment.size = 0.4) +
  labs(title="Fig 28 Correlatin graph for the second half of the book")+# repel =TRUE stops labe
ls overlapping; but also  attaches a segment line between node and text, which may be visually c
onfusing if it overlaps with graph edges and nodes
  theme_void()
```

Fig 28 Correlatin graph for the second half of the book



```
print('The diameter of the graph is')
```

```
## [1] "The diameter of the graph is"
```

```
print(diameter(correlation_graph))
```

```
## [1] 6
```

```
print("The radius of the graph is")
```

```
## [1] "The radius of the graph is"
```

```
print(radius(correlation_graph))
```

```
## [1] 3
```

```
print("The degrees are")
```

```
## [1] "The degrees are"
```

```
print(degree(correlation_graph))
```

```
##    lawyer     voice     house      time     light   cabinet     glass       set 
##         4        34        38        38        38        38        40        38 
##       sat      hand     hands    moment       day   strange       sir     night 
##        38        38        38        38        38        38        38        38 
##     henry     death    lanyon      mind     poole utterson    doctor       god 
##        38        44        44        44        38         4         2         2 
##    letter    edward      soul      body      evil    terror      dead 
##         4         2        10        10         8         6         8
```

```
print("The eccentricity is")
```

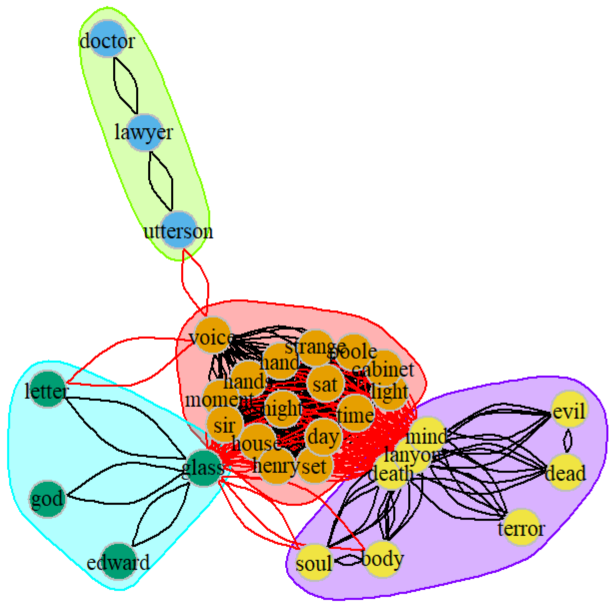
```
## [1] "The eccentricity is"
```

```
print(eccentricity(correlation_graph))
```

```
##    lawyer     voice     house      time     light   cabinet     glass       set 
##         5         3         4         4         4         4         5         4 
##       sat      hand     hands    moment       day   strange       sir     night 
##         4         4         4         4         4         4         4         4 
##     henry     death    lanyon      mind     poole utterson    doctor       god 
##         4         5         5         5         4         4         6         6 
##    letter    edward      soul      body      evil    terror      dead 
##         4         6         6         6         6         6         6
```

```
example.louv<- cluster_louvain(correlation_graph)
# membership vector
b<-membership(example.louv)


mod_max <- example.louv$modularity
k_opt <- length(example.louv)

# we can  use the plot.igraph approach to visualise the communities
plot(example.louv, correlation_graph, layout=lyt ,vertex.frame.color="gray", vertex.label.color=
"black", vertex.label.cex=0.7,main = paste("Fig 29.Louvain k=", k_opt, "; Modularity=", round(mo
d_max,2)))
```

## Fig 29.Louvain k= 4 ; Modularity= 0.13



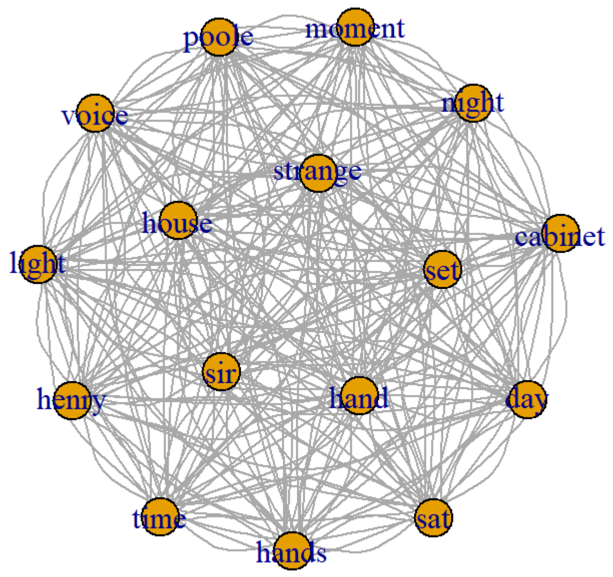
```
source("my_modularity.R")
source("cluster_measurements.R")
intra.cluster.df<-cluster_measurements(correlation_graph,example.louv)
knitr::kable(intra.cluster.df, align = 'l', caption = paste0(algorithm(example.louv),  " : size,
modularity and density scores of communities detected"))
```

multi level : size, modularity and density scores of communities detected

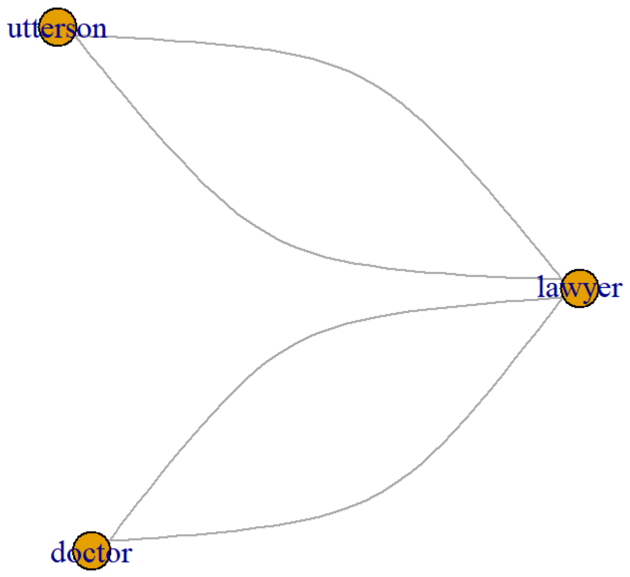
| cluster | size | modularity | density |
|---|---|---|---|
| -1 | 31 | 0.0000000 | 0.8989247 |
| 1 | 16 | 0.2444999 | 2.0000000 |
| 2 | 3 | 0.0094778 | 1.3333333 |
| 3 | 4 | 0.0141480 | 1.0000000 |
| 4 | 8 | 0.0865365 | 1.4285714 |

```
par(mfrow=c(1,1))
for(i in 1:length(unique(example.louv$membership))){
  plot(induced.subgraph(correlation_graph,example.louv$names[example.louv$membership==i]))
  title(paste0("Fig 30.",i,"louvian based community ",i))
}
```

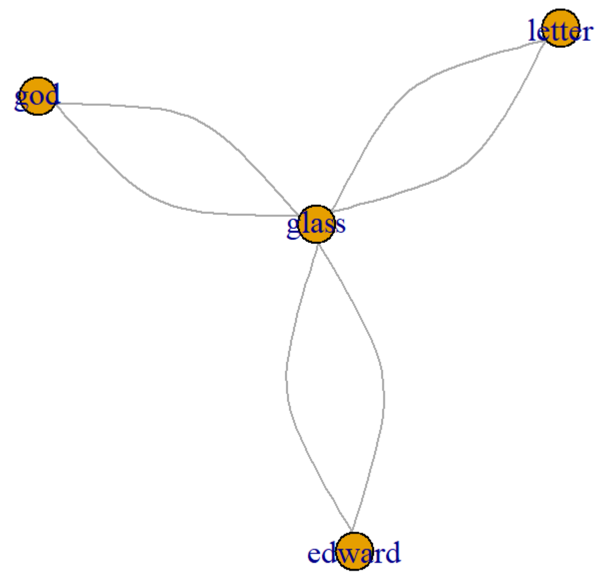
## Fig 30.1louvian based community 1



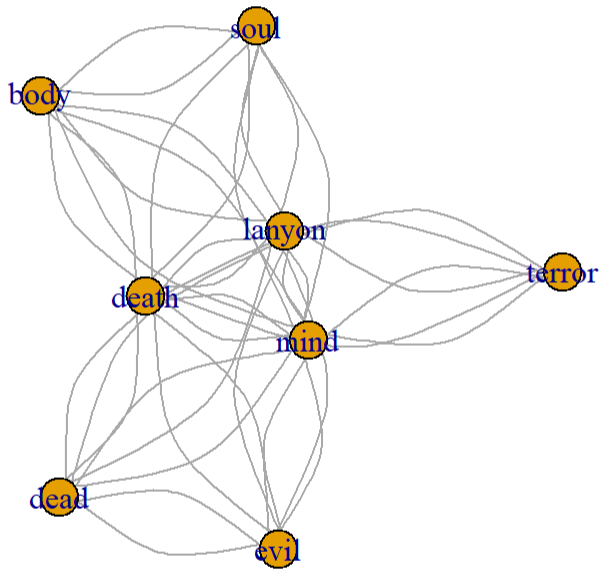
## Fig 30.2louvian based community 2



## Fig 30.3louvian based community 3



## Fig 30.4louvian based community 4



For the second half of the graph, the modularity is much low and the community is found to be 4.

The first community has strong intra connectivity and it tells the story involving poole,henry in a home at night time.

The second community is a much simple one which says utterson is a lawyer and has a relationship with a doctor.

The thrid community tells about the correlation between edward and his letter probably having connection with god or so.

The final community tells about the death persumably of lanyon and the evil terror sentiment that is present at that time.

From these, it is seen that the flow through present in the origianl book goes hand in hand with the story community tells us. It also again shows that the final chapter has the most negative sentiment and it is proven by the final community that has alot of negative scope in it.

Also, it is seen that the main character being a doctor comes throughout the story and the character poole also plays an important part from first half and second half. Both the half of the story talk about the friends of the main character and it clearly says that it means that friend is narrating a story about his friend’s life story which invloves death is again reconfirmed from the relationship between the halves.

## Conclusion

Through the text analysis, the insight from the book of The strange case of Dr.Jekyll and Mr.Hyde and the theme and relationships of the characters are found out successfully.

The overall theme of the book is found to be negative involving split personality of the Dr.Jekyll and death occurence of person.

The sentiment analysis and ngram approaches helped in identifying the moods of the words present and the important correlated words in the book which gave us better understanding.

When the summary is read, it is seen that the overall outline found using the community detection and the sentiment present in the book and the relationships present between the character are found to be similar thereby showing the importance of text analytics of the book.

It is still preferable to read the entire book in order to understand the actual meaning as it is only possible to get the outline based on text analysis.

## References

1.Web and Network science tutorials

2.Data Visualisation tutorials

3.https://www.tidytextmining.com/sentiment.html (https://www.tidytextmining.com/sentiment.html)

4.https://www.tidytextmining.com/ngrams.html (https://www.tidytextmining.com/ngrams.html)

5.https://www.rdocumentation.org/packages/ggraph/versions/1.0.2/topics/geom_node_text (https://www.rdocumentation.org/packages/ggraph/versions/1.0.2/topics/geom_node_text)