Social Network Analysis Assignment 1

Question 1

Read in the file "sample_generated_network.csv", located on Canvas. For the trust network, reproduce a plot similar to the one shown in class, showing the ties between individuals, their strength, and their direction. Exclude individuals who did not respond to the survey and who did not receive a complete set of responses about themselves from other students, but include everyone else.

Before importing the data, I inspect it at Excel. I found that two columns contains lots of blank values, thus I deleted them. Then I found the names of several columns don't appear at row names, then I deleted those columns. The final data contains 30 individuals.

```r
rm(list = ls(all = TRUE))
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 3.4.2
```

```
##
## Attaching package: 'igraph'
```

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

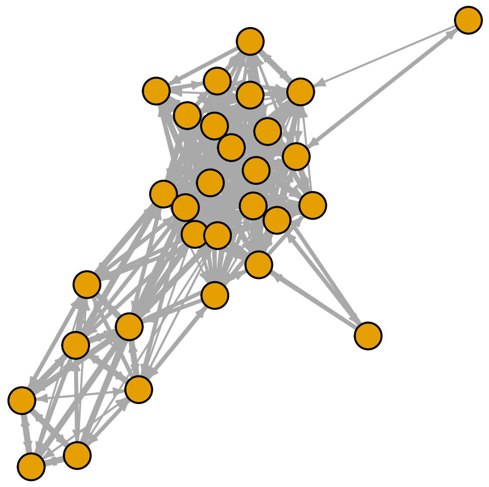
```
## The following object is masked from 'package:base':
##
##     union
```

```r
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.4.2
```

```r
setwd("C:/Users/cyma9/Dropbox/SocialNetwork/week1")
data = fread(file="sample_generated_network_new.csv", header = TRUE, stringsAsFactors=FALSE)

column_wise_replace = function(DT, x, y) {
    for(i in seq_along(x)){
         for (j in seq_len(ncol(DT))){
        set(DT,which(DT[[j]] == x[i]),j,y[i])
        }
    }
}




data = data[sample(nrow(data), replace = FALSE),]
advice = data[,(ncol(data) - 4):ncol(data)]
namekey = data[,1]

data = data[,2:(ncol(data) - 5)]

setcolorder(data, sample(colnames(data), replace = FALSE))
```

```
data = cbind(namekey, data, advice)


# make the choice data numeric
scale = cbind(c("Extremely Distrust  1","Distrust  2","Slightly Distrust  3","Neither Distrust Nor Trus

column_wise_replace(data, scale[,1], scale[,2])

# make adjacency matrix

# subset to just the trust choices, then sort on the columns and rows to make each entry match up
# data are directed, so matrix will not be symmetric
adj = as.data.frame(data[,1:(ncol(data) - 5)])
rownames(adj) = adj[,1]
adj = adj[,-1]

adj = adj[sort(rownames(adj)),sort(colnames(adj))]

trusting = adj
trusting[trusting==-3] = 0
trusting[trusting==-2] = 0
trusting[trusting==-1] = 0

trusting_graph = graph.adjacency(as.matrix(trusting), "directed", weighted = TRUE)


plot.igraph(trusting_graph,vertex.label=NA,layout=layout.fruchterman.reingold, vertex.label.color="blac
```

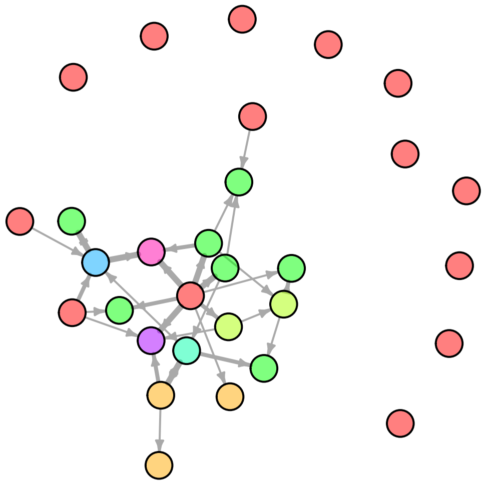


Question 2

Repeat the same as in Question 1, but for the distrust network. For this plot, color the nodes ordinally to correspond to how distrusted they are (check rainbow or colorRampPalette). Exclude individuals who did not respond to the survey and who did not receive a complete set of responses about themselves from other students, but include everyone else.

```r
distrust = adj
distrust[distrust == 3] = 0
distrust[distrust == 2] = 0
distrust[distrust == 1] = 0
distrust[distrust == -3] = 3
distrust[distrust == -2] = 2
distrust[distrust == -1] = 1
distrust_num = apply(distrust,2,as.numeric)
dev = colSums(distrust_num)
pal <- rainbow(max(dev) - min(dev) + 1 , alpha=.5)

distrust_graph = graph.adjacency(as.matrix(distrust_num), "directed", weighted = TRUE)
col = rep(0,30)
for ( i in c(1:30))
{
  col[i] = pal[dev[i] + 1]
}
```

```
plot.igraph(distrust_graph,vertex.label=NA,layout=layout.fruchterman.reingold, vertex.color = col, vert
```



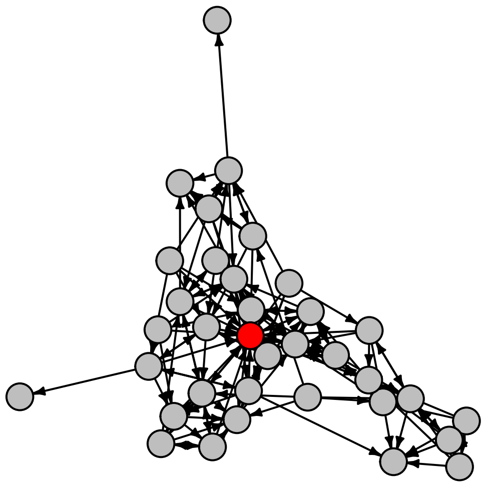
Question 3

Generate a plot for the advice network showing the ties between individuals and their direc-tion. For this plot, indentify with a color the individual to whom the most people go to for advice.

```
count = rep(0,38)

advice_edge1 = cbind(data[,1], data$V38)
for(i in c(1:30))
  {
  count[data$V38[i]] =  count[data$V38[i]] + 1
  }

advice_edge2 = cbind(data[,1], data$V39)
for(i in c(1:30))
{
  count[data$V39[i]] =  count[data$V39[i]] + 1
}

advice_edge3 = cbind(data[,1], data$V40)
for(i in c(1:30))
{
  count[data$V40[i]] =  count[data$V40[i]] + 1
}

advice_edge4 = cbind(data[,1], data$V41)
for(i in c(1:30))
{
  count[data$V41[i]] =  count[data$V41[i]] + 1
}

advice_edge5 = cbind(data[,1], data$V42)
for(i in c(1:30))
{
  count[data$V42[i]] =  count[data$V42[i]] + 1
}
print(count)
```

```
##  [1]  3  7  0  6  4  1  0 19  2  5  4  2  1  3  8  4  5  0  7  6  5  3  2
## [24]  2  0  2 11  2  2  3  6  0  5  7  5  8  0  0
```

```
advice_edges = rbind(advice_edge1, advice_edge2, advice_edge3, advice_edge4, advice_edge5)

advice_seeking = graph.data.frame(advice_edges, directed = TRUE)
for ( i in c(1:35))
{
  V(advice_seeking)$color[i] = 'grey'
  if (V(advice_seeking)$name[i] == which.max(count))
  {
    V(advice_seeking)$color[i] = 'red'
  }
}

plot.igraph(advice_seeking,vertex.label=NA,layout=layout.fruchterman.reingold, vertex.label.color="blac
```



The point with red color is the individual to whom the most people go to for advice.

Question 4

How many of the relationships are reciprocated in each of the three networks? Do you think that this is more or less than would be reciprocated by random chance? Treat the trust and distrust relationships as binary. Perform this calculation directly on the matrices.

```
# trust
count_trust = 0
total_trust = 0
for (i in c(1:30))
{
  for (j in c(1:30))
  {
    if (adj[i,j]>0)
      total_trust = total_trust + 1
    if ((adj[i,j]>0)&(adj[j,i]>0)){
      count_trust = count_trust + 1

    }
  }
}

count_distrust = 0
```

```r
total_distrust = 0

for (i in c(1:30))
{
  for (j in c(1:30))
  {
    if (adj[i,j]<0)
      total_distrust = total_distrust + 1
    if ((adj[i,j]<0)&(adj[j,i]<0))
      count_distrust = count_distrust + 1
  }
}


advice_adj = matrix(0,nrow = 37, ncol = 37)
for ( i in c(1:nrow(advice_edges)))
{
  x = advice_edges[i]$`To begin, please select your name.`
  y = advice_edges[i]$V2
  advice_adj[x,y] = 1
}
total_advice = 0
count_advice = 0
for ( i in c(1:37))
{
  for ( j in c(1:37))
  {
    if (advice_adj[i,j]==1){
      total_advice = total_advice + 1}

    if ((advice_adj[i,j]==1)&(advice_adj[j,i]==1))
      count_advice = count_advice + 1
  }
}

print('How many of the trust relationships are reciprocated?')
```

```
## [1] "How many of the trust relationships are reciprocated?"
```

```r
print(count_trust / 2)
```

```
## [1] 152
```

```r
print('How many of the distrust relationships are reciprocated?')
```

```
## [1] "How many of the distrust relationships are reciprocated?"
```

```r
print(count_distrust / 2)
```

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

```r
print('How many of the seeking advice relationships are reciprocated?')
```

```
## [1] "How many of the seeking advice relationships are reciprocated?"
```

```r
print(count_advice / 2)
```

```
## [1] 33
```

```
prob_trust = total_trust/900
prob_distrust = total_distrust/900
prob_advice = total_advice/900

print('If by random chance: ')
```

```
## [1] "If by random chance: "
```

If by random chance:

```
print('How many of the trust relationships are reciprocated?')
```

```
## [1] "How many of the trust relationships are reciprocated?"
```

```
print(prob_trust * prob_trust * 450)
```

```
## [1] 85.805
```

```
print('How many of the distrust relationships are reciprocated?')
```

```
## [1] "How many of the distrust relationships are reciprocated?"
```

```
print(prob_distrust * prob_distrust * 450)
```

```
## [1] 0.5338889
```

```
print('How many of the seeking advice relationships are reciprocated?')
```

```
## [1] "How many of the seeking advice relationships are reciprocated?"
```

```
print(prob_advice * prob_advice * 450)
```

```
## [1] 12.5
```

It's more than by random chance

Question 5

How many of the relationships in the trust network also exist in the advice network? How many of the relationships in the distrust network also exist in the advice network? Do you think this is surprising? Exclude individuals who did not respond to the survey and who did not receive a complete set of responses about themselves from other students, but include everyone else. Treat the trust and distrust relationships as binary. Perform this calculation directly on the matrices.

```
count_advice_trust = 0
count_advice_distrust = 0

trust_with_advice = trusting
trust_without_advice = trusting

distrust_with_advice = distrust
distrust_without_advice = distrust

advice_with_no_feeling = trusting
```

```r
for (i in c(1:30))
  for (j in c(1:30))
    advice_with_no_feeling[i,j] = 0


for (i in c(1:30))
{
  for (j in c(1:30))
  {
    x = as.numeric(colnames(adj)[i])
    y = as.numeric(rownames(adj)[j])
    if ((adj[i,j]>0) & (advice_adj[x,y]>0))
    {
      count_advice_trust = count_advice_trust + 1
      trust_without_advice[i,j] = 0
    }
    if ((adj[i,j]>0) & (advice_adj[x,y] == 0))
    {
      trust_with_advice[i,j] = 0
    }
    if ((adj[i,j]<0) & (advice_adj[x,y]>0))
    {
      count_advice_distrust = count_advice_distrust + 1
      distrust_without_advice[i,j] = 0
    }

     if ((adj[i,j]<0) & (advice_adj[x,y] == 0))
    {
      distrust_with_advice[i,j] = 0
     }
    if ((adj[i,j] == 0) & (advice_adj[x,y] == 1))
    {
      advice_with_no_feeling[i,j] = 1
    }

  }
}
print('How many of the relationships in the trust network also exist in the advice network?')
```

```
## [1] "How many of the relationships in the trust network also exist in the advice network?"
```

```r
print(count_advice_trust)
```

```
## [1] 121
```

```r
print('How many of the relationships in the distrust network also exist in the advice network?')
```

```
## [1] "How many of the relationships in the distrust network also exist in the advice network?"
```

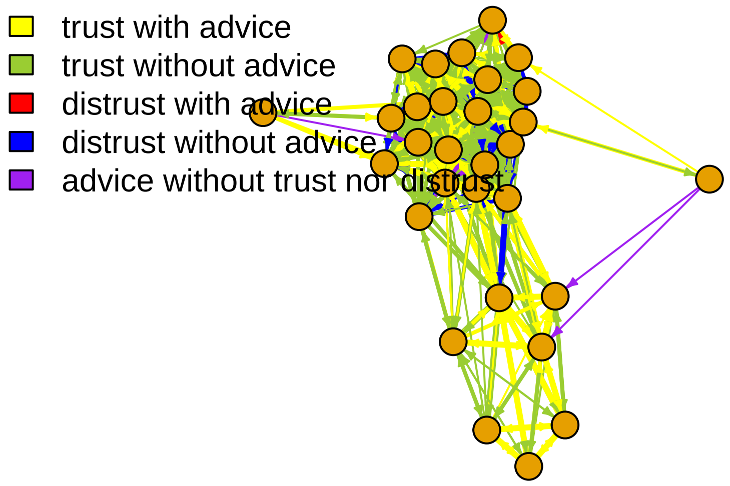
```r
print(count_advice_distrust)
```

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

Extra Question

Making use of the calculations from Question 5, generate a plot for both the trust and distrust networks that also shows the advice relationships. Distinguish on the plot between ties that overlap, as well as distinguish between advice ties and trust and distrust ties, and leave the strength of the trust and distrust relationships independent from the presence of an advice tie (do not treat the trust and distrust relationships as binary). Exclude individuals who did not respond to the survey and who did not receive a complete set of responses about themselves from other students, but include everyone else.

```
trust_with_advice_graph = graph.adjacency(as.matrix(trust_with_advice), "directed", weighted = TRUE)
trust_without_advice_graph = graph.adjacency(as.matrix(trust_without_advice), "directed", weighted = TRU
E(trust_with_advice_graph)$color = 'yellow'
E(trust_without_advice_graph)$color = 'yellowgreen'
union_trust = union(trust_with_advice_graph, trust_without_advice_graph)
E(union_trust)$color <- ifelse(is.na(E(union_trust)$color_1),E(union_trust)$color_2,E(union_trust)$colo
E(union_trust)$weight <- ifelse(is.na(E(union_trust)$weight_1),E(union_trust)$weight_2,E(union_trust)$we

distrust_with_advice_graph = graph.adjacency(as.matrix(distrust_with_advice), "directed", weighted = TRU
distrust_without_advice_graph = graph.adjacency(as.matrix(distrust_without_advice), "directed", weighte
E(distrust_with_advice_graph)$color = 'red'
E(distrust_without_advice_graph)$color = 'blue'
union_distrust = union(distrust_with_advice_graph, distrust_without_advice_graph)
E(union_distrust)$color <- ifelse(is.na(E(union_distrust)$color_1),E(union_distrust)$color_2,E(union_di
E(union_distrust)$weight <- ifelse(is.na(E(union_distrust)$weight_1),E(union_distrust)$weight_2,E(union_



union_final = union(union_trust, union_distrust)
E(union_final)$color <- ifelse(is.na(E(union_final)$color_1),E(union_final)$color_2,E(union_final)$colo
E(union_final)$weight <- ifelse(is.na(E(union_final)$weight_1),E(union_final)$weight_2,E(union_final)$we

advice_with_no_feeling_graph = graph.adjacency(as.matrix(advice_with_no_feeling), "directed", weighted =
E(advice_with_no_feeling_graph)$color = 'purple'
union_grand_final = union(union_final, advice_with_no_feeling_graph)
E(union_grand_final)$color <- ifelse(is.na(E(union_grand_final)$color_1),E(union_grand_final)$color_2,E
E(union_grand_final)$weight <- ifelse(is.na(E(union_grand_final)$weight_1),E(union_grand_final)$weight_

plot.igraph(union_grand_final,vertex.label=NA,layout=layout.fruchterman.reingold, vertex.label.color="bl
legend('topleft',fill = c('yellow','yellowgreen','red','blue','purple'),
     legend=c('trust with advice','trust without advice','distrust with advice', 'distrust without advi
```


🟨 trust with advice
🟩 trust without advice
🟥 distrust with advice
🟦 distrust without advice
🟪 advice without trust nor distrust