MATPMD2 Assignment 2019

2834777

Introduction

The network considered in this analysis is a network of interactions of bottlenose dolphins active in Cedar Key, Florida. The network data was collected by Stefanie Gazda, Swami Iyer, Timothy Killingback, Richard Connor and Solange Brault and released for their 2015 paper "The importance of delineating networks by activity type in bottlenose dolphins (Tursiops truncatus) in Cedar Key, Florida (Gazda et all, 2019).

The data in its raw form is a record of individual sightings of dolphins which includes the time and date of sighting, the individual dolphins sighted, and the activity the sighted dolphins were engaged in at the time. The data for this network was found at The Network Repository but is also available from the Dryad Digital repository.

Bottlenose dolphins are highly intelligent creatures whose lives are mysterious because they occur beneath the sea which is an alien environment for most people. This network data is interesting because it gives an insight into those lives. Apart from giving an idea of how sociable dolphins are through simply how many individuals each interacts with, the data has specifically been split by activity type giving three subnetworks allowing an insight into how dolphin-to-dolphin interactions are altered when the animals are engaged in explicitly social behaviour, when they are foraging, and when they are travelling.

The network analysed is a social network so this work will consider how connected each node (dolphin) is, and how clustered the network is. These questions will be asked for the overall graph and for the social, forage, and travel subgraphs separately. The size and connectivity of the networks will be considered, along with the overlap of the three subnetworks, to ask how complete the network could be.

Methods

Nodes in the network analysed here are individual bottlenose dolphins sighted in Cedar Key, Florida. An edge is included in the graph if the two incident nodes/dolphins are sighted in each other's company. This is also true of the separate social, forage, and travel subgraphs except nodes are only included if the dolphins are sighted engaged in each category of activity. Similarly, dolphins must be seen together, engaged in the categories of data for an edge to be included in the three subgraphs.

The size of the overall network and of each of the three subnetworks will be described by determining the number of nodes and edges in each.

The connectivity of the networks will be described by considering the number of components in each graph, and by considering measures of diameter and radius.

Network clustering will be described by determining cluster coefficients for nodes and for graphs, and by determining the number and size of cliques within each graph.

The importance of individual nodes will be considered by determining the degree of each node and a measure of betweenness centrality, closeness centrality, and degree centrality for each node. Distribution of these measures throughout the graph will considered. Highly central individuals will be identified.

Finally, the frequency with which each dolphin occurs in the raw data, and the overlap between the social, forage, and travel subnetworks will be considered with regards to completeness of the data collection and so the usefulness of any network analysis.

Results

Network Size

- The overall network of dolphins active in Cedar Key, Florida, contains 291 nodes and 3182 edges.
- The network of dolphins active socially in Cedar Key, Florida, contains 151 nodes and 1554 edges.
- The network of dolphins actively foraging in Cedar Key, Florida, contains 190 nodes and 1134 edges.
- The network of dolphins actively travelling in Cedar Key, Florida, contains 188 nodes and 1032 edges.

Connectivity

- The overall network of dolphins consists of 3 disconnected components. A primary component of 284 nodes, and two secondary components of 5 and 2 nodes.
- The network of dolphins active socially consists of 2 disconnected components A primary component of 149 nodes, and single secondary component of just 2 nodes.
- The network of dolphins actively foraging consists of 6 disconnected components. A primary component of 176 nodes, and five secondary components of 4, 3, 3, 2, and 2 nodes.
- The network of dolphins actively travelling consists of 6 disconnected components. A primary component of 157 nodes, a secondary component of 17 nodes, and four secondary components of 5, 4, 3, and 2 nodes.

For each of the four networks considered, the primary component contained the vast majority of the graph: for the least connected graph, the network of actively travelling dolphins, the primary component still contained 84% of all nodes.

Measures of diameter and radius of the primary component of each graph were calculated. Diameter is the maximum graph eccentricity and radius is the minimum, both give an indication of the distances across graphs.

GRAPH	DIAMETER	RADIUS
OVERALL	6	4
SOCIAL	4	2
FORAGE	7	4
TRAVEL	7	4

Clustering

- 34% (103 of 291) of nodes in the overall network have a cluster coefficient of 1. The average for the graph is 0.682.
- 47% (71 of 151) of nodes in the social network have a cluster coefficient of 1. The average for the graph is 0.788.
- 41% (77 of 190) of nodes in the forage network have a cluster coefficient of 1. The average for the graph is 0.659.
- 47% (88 of 188) of nodes in the travel network have a cluster coefficient of 1. The average for the graph is 0.728.

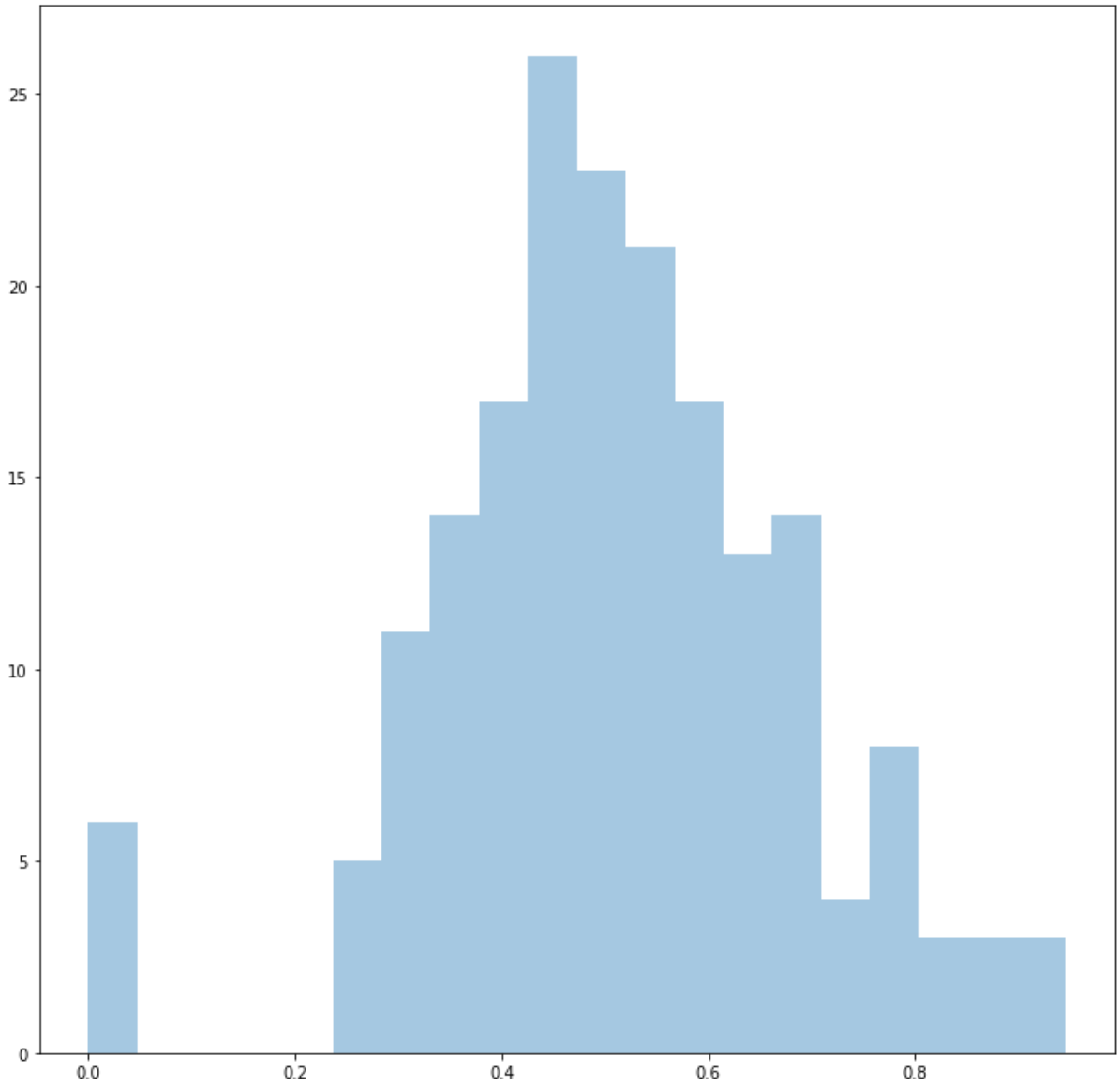


Figure 1 - Distribution of cluster coefficients for Overall network (excludes values of 1)

The distribution of cluster coefficients in the overall network is shown in figure 2. Excluding a large number of values of 1, and a handful of values of 0, the data appears broadly symmetrical and unimodal, ranging from 0.2 to 0.9. Gaps exist between the bulk of the data and the excluded values of 1 and 0. The distribution is similar for the social, forage, and travel subnetworks.

- The overall network contains 941 maximal cliques (from 291 nodes), the largest of which consists of 31 nodes.
- The social network contains 116 maximal cliques (from 151 nodes), the largest of which consists of 31 nodes.
- The forage network contains 136 maximal cliques (from 190 nodes), the largest of which consists of 18 nodes.

- The travel network contains 131 maximal cliques (from 188 nodes), the largest of which consists of 16 nodes.

Individual Significance

- The overall network has a maximum node degree of 99, a mean node degree of 21.9, and a median node degree of 15.
- The social network has a maximum node degree of 69, a mean node degree of 20.6, and a median node degree of 15.
- The forage network has a maximum node degree of 47, a mean node degree of 11.9, and a median node degree of 8.
- The travel network has a maximum node degree of 43, a mean node degree of 11.0, and a median node degree of 9.

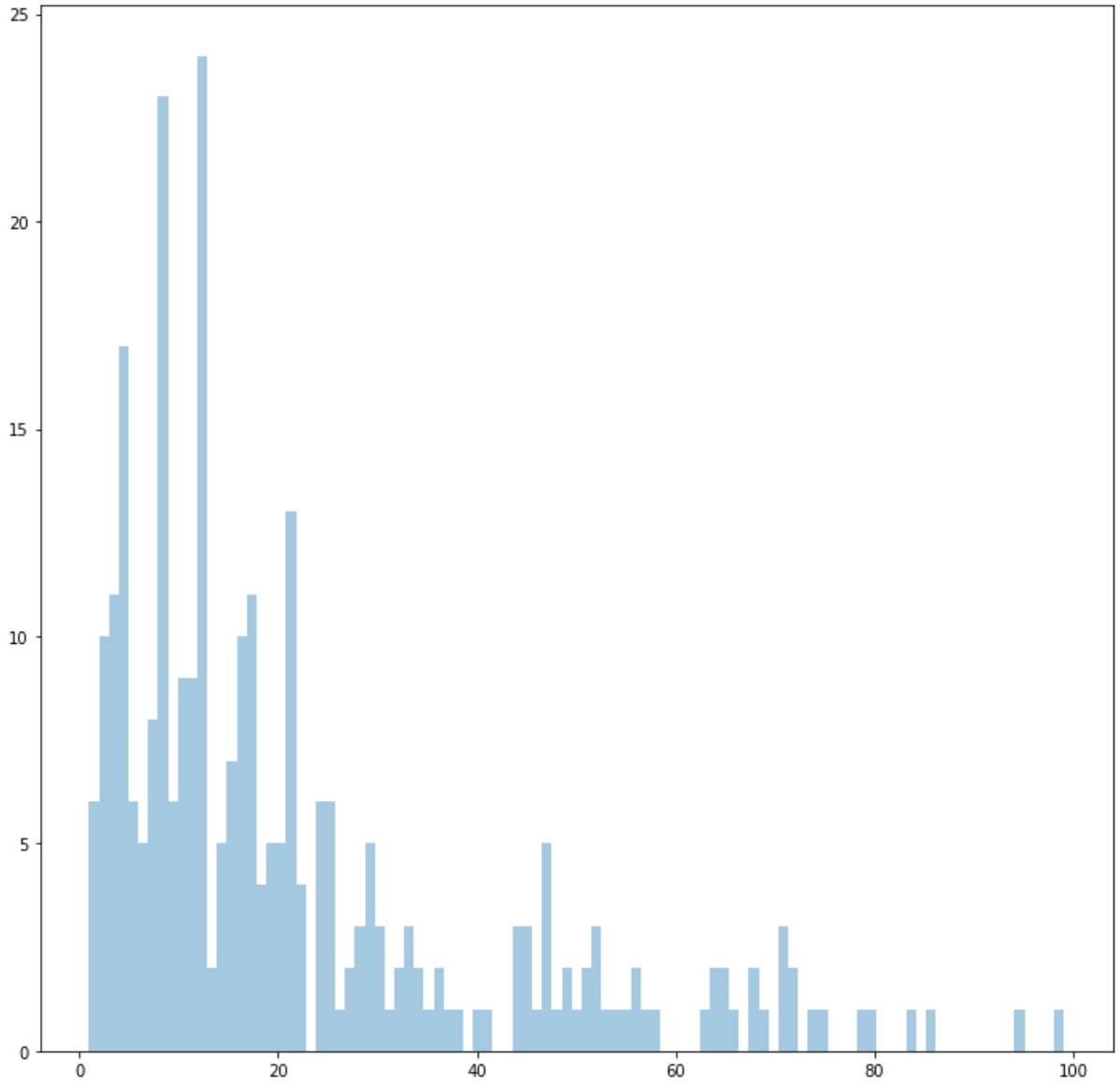


Figure 2 - Overall network node degree distribution

The distribution of node degree in the overall network appears to be broadly unimodal and significantly positively skewed. The data ranges from 1 to 99 with large gaps at higher values though no apparent outliers. This pattern is repeated for distributions of node degree of the social, forage, and travel subnetworks.

The measures of closeness centrality, betweenness centrality, and degree centrality have been calculated for each node in the overall network.

- Closeness centrality values are largely distributed around 0.4 with a significant negative skew.
- Betweenness centrality values are largely distributed between 0 and 0.005 with a significant positive skew pushing the maximum as high as 0.1.
- Degree centrality values are distributed between 0.025 and 0.1 with a positive skew pushing the maximum as high as 0.35

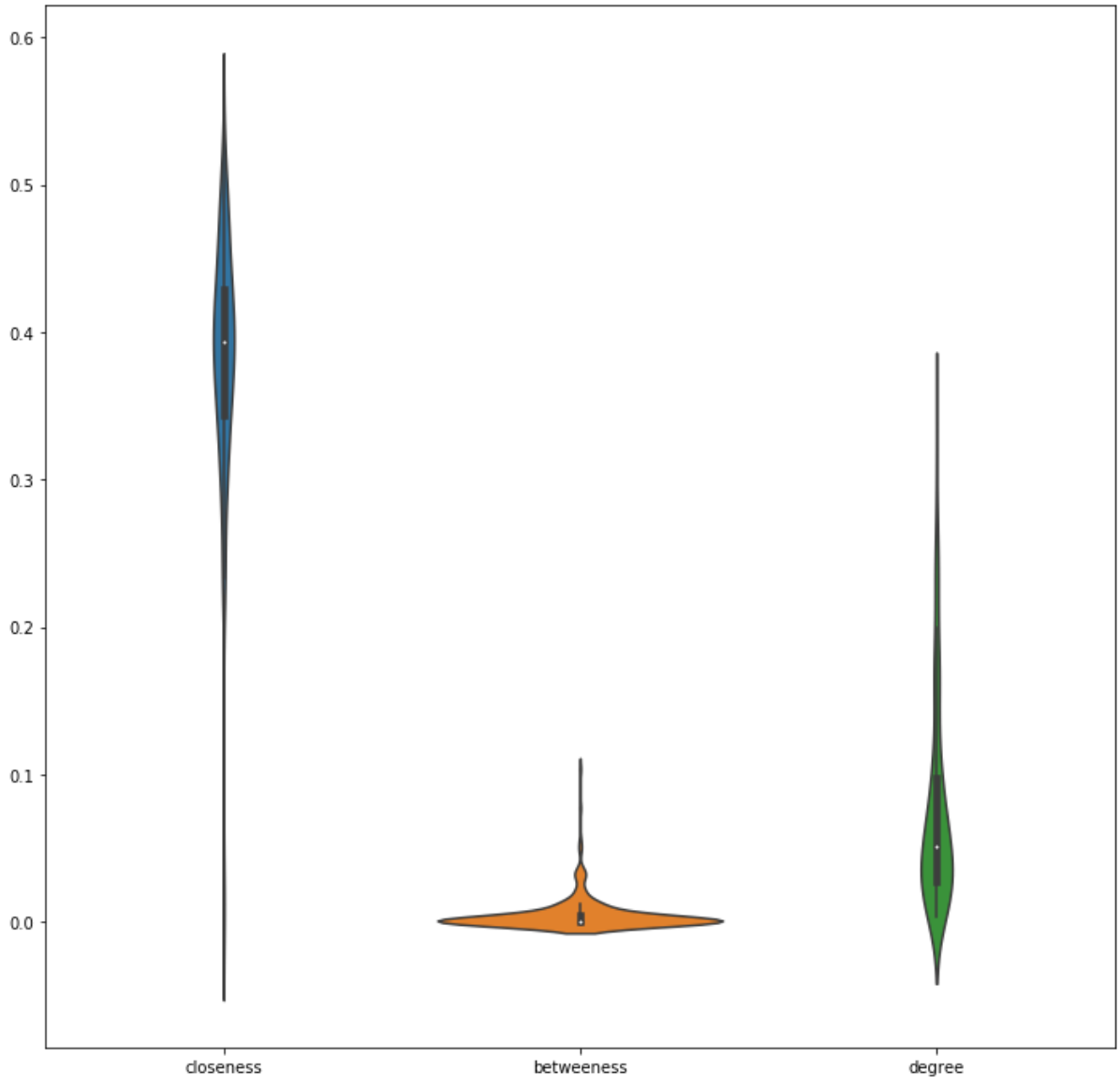


Figure 3 - Violin plot of closeness centrality, betweenness centrality, and degree centrality for nodes in overall network

Five Figure Summaries of Overall Network Centrality Measures					
Measure	Min	Q1	Median	Q3	Max
Closeness	0.00344828	0.34392203	0.39340308	0.42916726	0.53314472
Betweenness	0.0	0.0	0.00085667	0.00523156	0.10392713
Degree	0.00344828	0.02758621	0.05172414	0.09827586	0.34137931

Table 1- Five Figure Summaries of Overall Network Centrality Measures

The nodes in the overall network were all ranked 1 to 291 (high to low) for each calculated measure of centrality. The 5 with the highest combined ranking across the three measures, as well as the highest rank for each measure were combined into a list. This created a list of 6 highly central nodes (two were both highest ranked for a specific measure and included in the 5 ranked highest overall).

Centrality Measures for Select Highly Central Nodes							
Node	closeness	betweeness	degree	c_rank	b_rank	d_rank	Tot_rank
CRSC	0.5221	0.0561	0.3414	2	3	1	1
OLWB	0.5331	0.0363	0.3276	1	6	2	2
SOCT	0.5143	0.0511	0.2897	4	4	4	3
LKWB	0.5049	0.0339	0.2759	6	10	5	4
MHWB	0.5172	0.0224	0.2966	3	18	3	5
KKNK	0.4203	0.1039	0.0724	88	1	103	58

Table 2 - Centrality Measures for Select Highly Central Nodes

These nodes and their edges are shown within the overall network in figure 4 below.

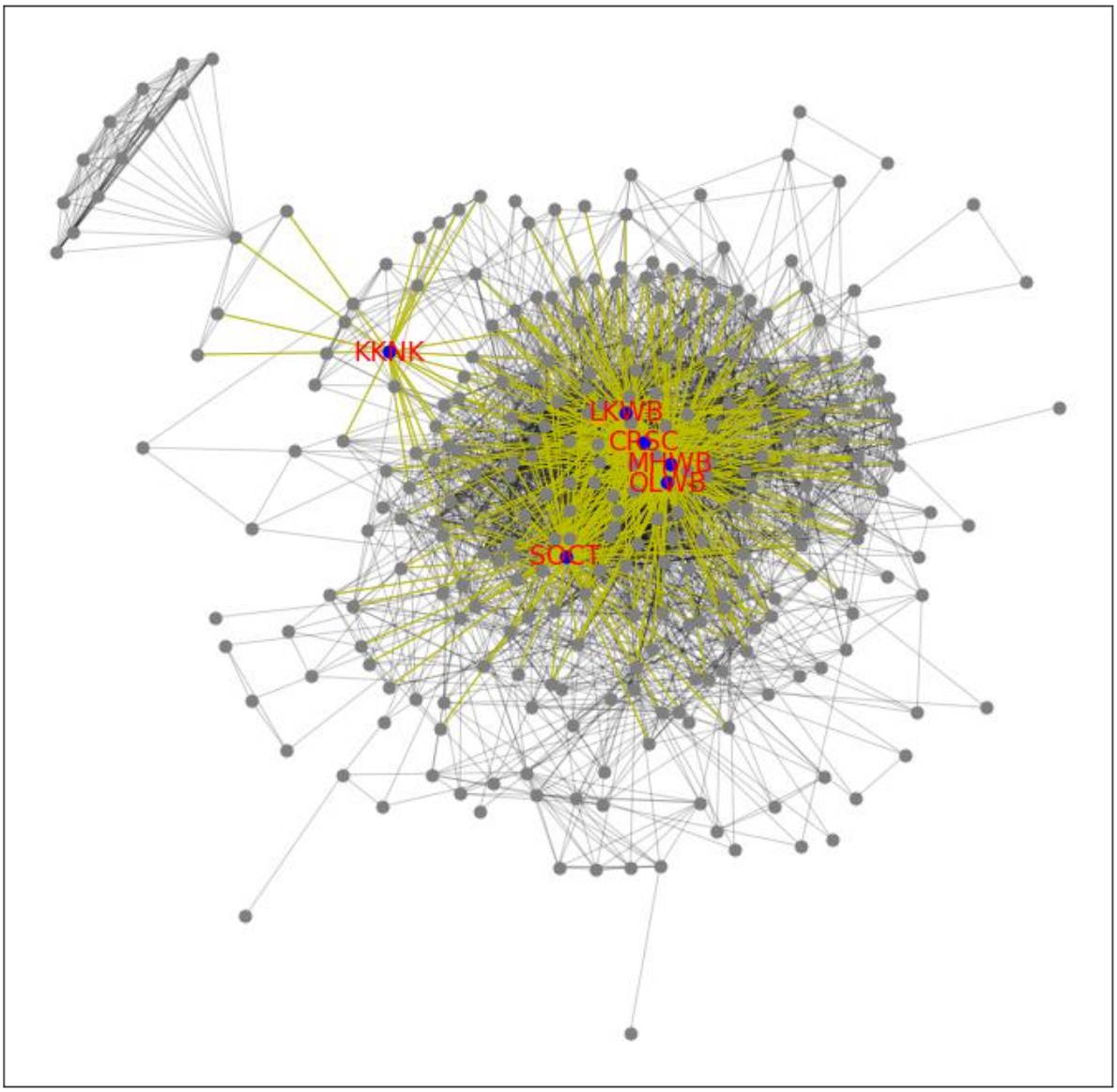


Figure 4 -Overall Network with highly central nodes highlighted

Network Completeness

The distribution of the count for each dolphin in the raw data is shown in figure 5 below. Figure 5 shows that the count data is heavily concentrated towards low values, with a large number of dolphins counted only once. The median and mean for the count have been determined as 3 and 4.364 respectively. The dolphins CRSC, SOCT, OLWB, LKWB, MHWB, and KKNK are counted 20, 18, 20, 11, 18, and 3 times respectively.

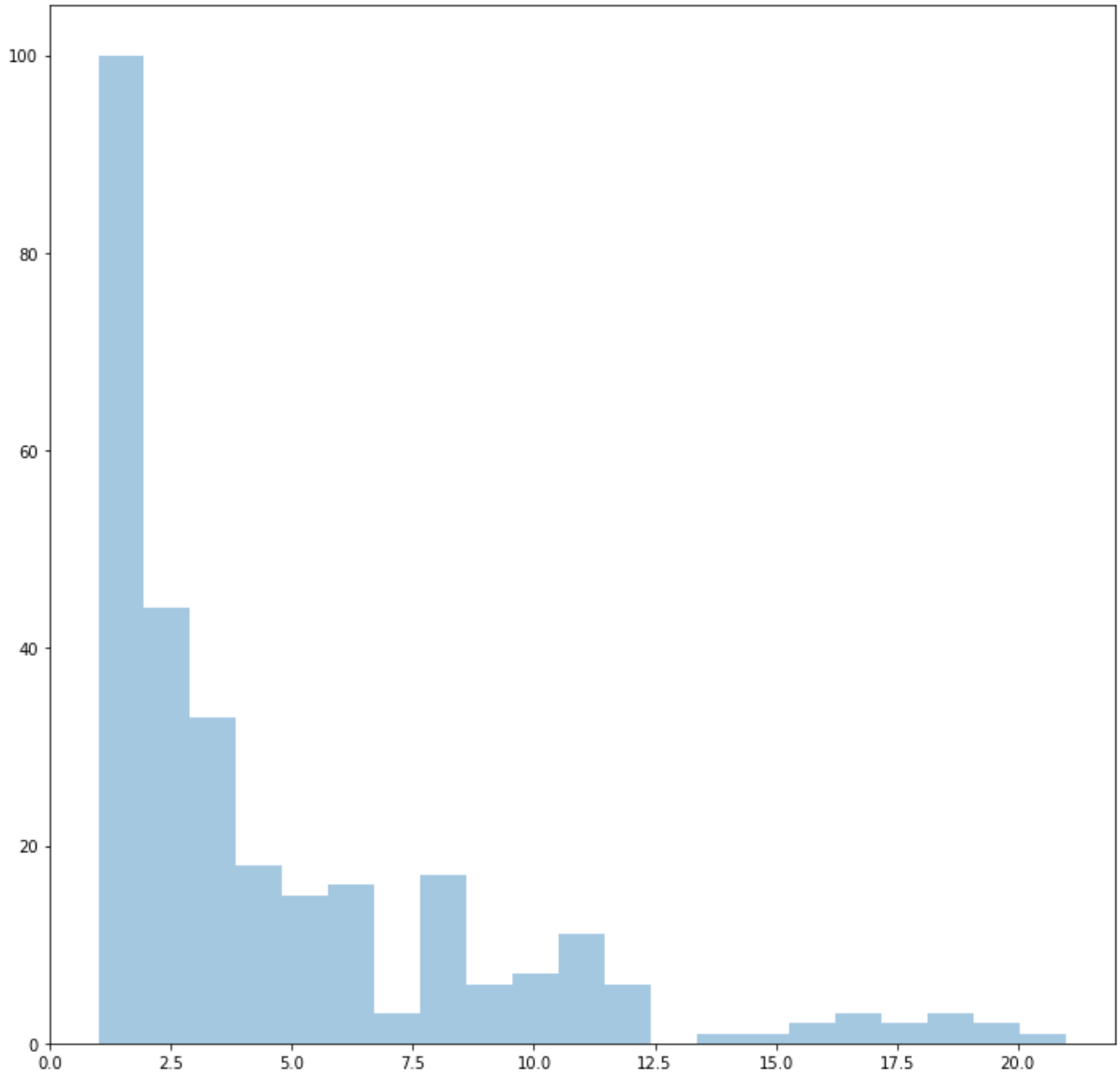


Figure 5 - Distribution of count of dolphins in network raw data

A Venn diagram of the three subnetworks has been produced and is shown in figure 6 below. It shows that of the nodes in the network (observed dolphins) only 81 are contained within the social, forage, and travel subnetworks; in contrast, 134 nodes are contained within only one of the three networks.

All 6 of the list of highly central dolphins are included in all three of the subnetworks meaning each of the highly central dolphins has been observed interacting socially, while foraging, and while traveling with all other dolphins they have interacted with.

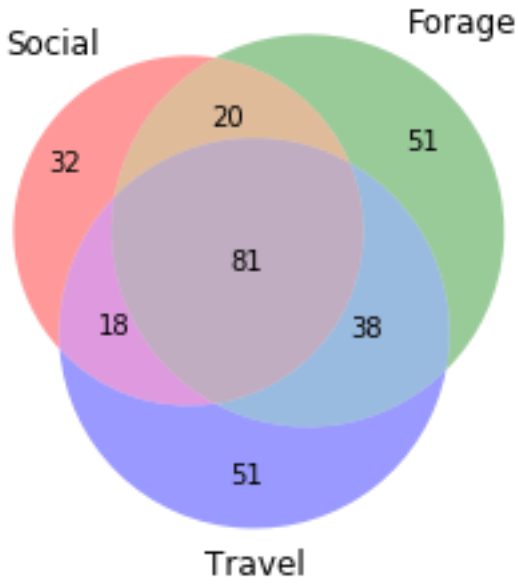


Figure 6 - Venn diagram of social, forage, and travel subnetworks

Conclusion

Network Size

The size metrics of the network suggest a large population of dolphins: approximately 300 individuals. With over 3000 edges, on average, each individual is connected to over 20 others which is suggestive of a highly connected group.

It is notable that the edge to node ratio is significantly lower for the forage and travel subnetworks that for the social subnetwork (6.0, 5.5, and 10.3 respectively). This suggests that these dolphins interact more widely while socializing than during foraging and travel activities.

Connectivity

That the network is nearly complete with a primary component containing the vast majority of nodes and only small disconnected splinter components suggests that the researchers who gathered this data were looking at a single population rather than a number of entirely separate groups.

Again, that the social subnetwork has only two components while the forage and travel subnetworks each have 6 disconnected components suggests that dolphins associate more widely while socialising than while engaged in other activity.

This view is also backed by the fact that both graph diameter and radius are larger for the forage and travel subnetworks than for the social subnetwork indicating shorter path lengths in the social subnetwork.

Clustering

The networks are all highly clustered. A large number of nodes have a cluster coefficient of 1 indicating that all their connections are connected to each other. Nearly half of the nodes in the social and travel subnetworks have cluster coefficients of 1, as do over a third of the overall network's nodes.

More broadly, the average cluster coefficients for all graphs are high, between 0.659 and 0.788, reflecting the fact this is a network of social beings.

The networks also contain a lot of cliques, nearly 1000 for the overall network, some of which are very large. Including 31 nodes, the largest clique in the social subnetwork includes over 20% of the dolphins in that network.

Again, it is notable that the largest clique in the social subnetwork is significantly larger than the largest cliques in the forage and social subnetworks (31, 18, and 16 respectively). This indicates wider association in the social network.

Interestingly the number of cliques in the social subnetwork is smaller than in the forage and social subnetworks (116, 136, and 131 respectively). This might indicate that the social subnetwork is closer to being a complete graph than the forage and social subnetworks as a complete graph has only one clique.

Individual Significance

Even within this highly connected graph a few nodes stand out as being especially central. The dolphins CRSC, SOCT, OLWB, LKWB, and MHWB stand out for having high values for closeness centrality, betweenness centrality, and degree centrality. These could be the most dominant dolphins in the population.

The dolphin KKNK ranks modestly for both closeness centrality and degree centrality but forms the only observed link between a group of 16 dolphins and the rest of the population so scores highly on betweenness centrality. It is possible that this is actually a separate population and that the researchers have observed the dolphin KKNK moving from one group to another, either temporarily or permanently.

Summary

As is, the network data analysed indicates that dolphins are generally highly connected animals with a few especially central individuals; and that dolphins associate most widely while socializing but keep to smaller groups while foraging and while traveling.

Analysis Validity

The data available at the Network Repository included the overall network and the three subnetworks but, for some reason, the dolphin "names" were replaced with the numerals 1,

2, 3, etc and the four networks were treated separately so it couldn't be established which dolphin from the overall network was in which of the subnetworks. The original data, available from the Dryad Digital repository was used to re-establish this link.

Three measures of the network(s) suggest that the data provided is an incomplete picture of the population of dolphins considered:

1. The edges of the network are included if the two incident dolphins are observed in each other's company. This means that a cluster coefficient of 1 would result from an individual dolphin only being observed once. At an isolated single observation the observed dolphins would form the nodes of a complete graph and each would have a cluster coefficient of 1. The value of this coefficient would be reduced if any of the dolphins was again observed, but with a different group because there would be only limited links between the two groups. That so many dolphins (103 of 291) have a cluster coefficient of 1 suggests that many of these dolphins have only been observed once. This can be confirmed by reviewing the raw data which shows that 100 dolphins were only observed once.
2. The observed dolphins CRSC, SOCT, OLWB, LKWB, and MHWB, stand out as especially connected within the network. However, these 5 dolphins have been observed 20, 18, 20, 11, and 18 times, respectively, where the median and mean for the population are 3 and 4.364 respectively. This suggests that the stand-out dolphins have been observed in the company with lots of other dolphins, not because they are especially connected but because they have been observed lots of times.
3. The social, forage, and travel subnetworks overlap only partially: of the 291 dolphins in the overall network only 81 are contained within the social, forage, and travel subnetworks while 134 nodes are contained within only one of the three networks. In contrast the social, forage, and travel neighbourhood networks of all six of the identified highly central dolphins overlap perfectly. Additionally, it appears implausible that any dolphins are only engaged in only social, or forage, or travel activity.

These three measures indicate that the researchers have only been able to create a partial picture of the lives of these dolphins. For this reason, conclusions drawn from the analysis of these networks is of questionable value.

References

Gazda S, Iyer S, Killingback T, Connor R, Brault S. 2015 The importance of delineating networks by activity type in bottlenose dolphins (Tursiops truncatus) in Cedar Key, Florida.R. Soc. open sci.2: 140263. http://dx.doi.org/10.1098/rsos.140263

Appendix 1 – Code

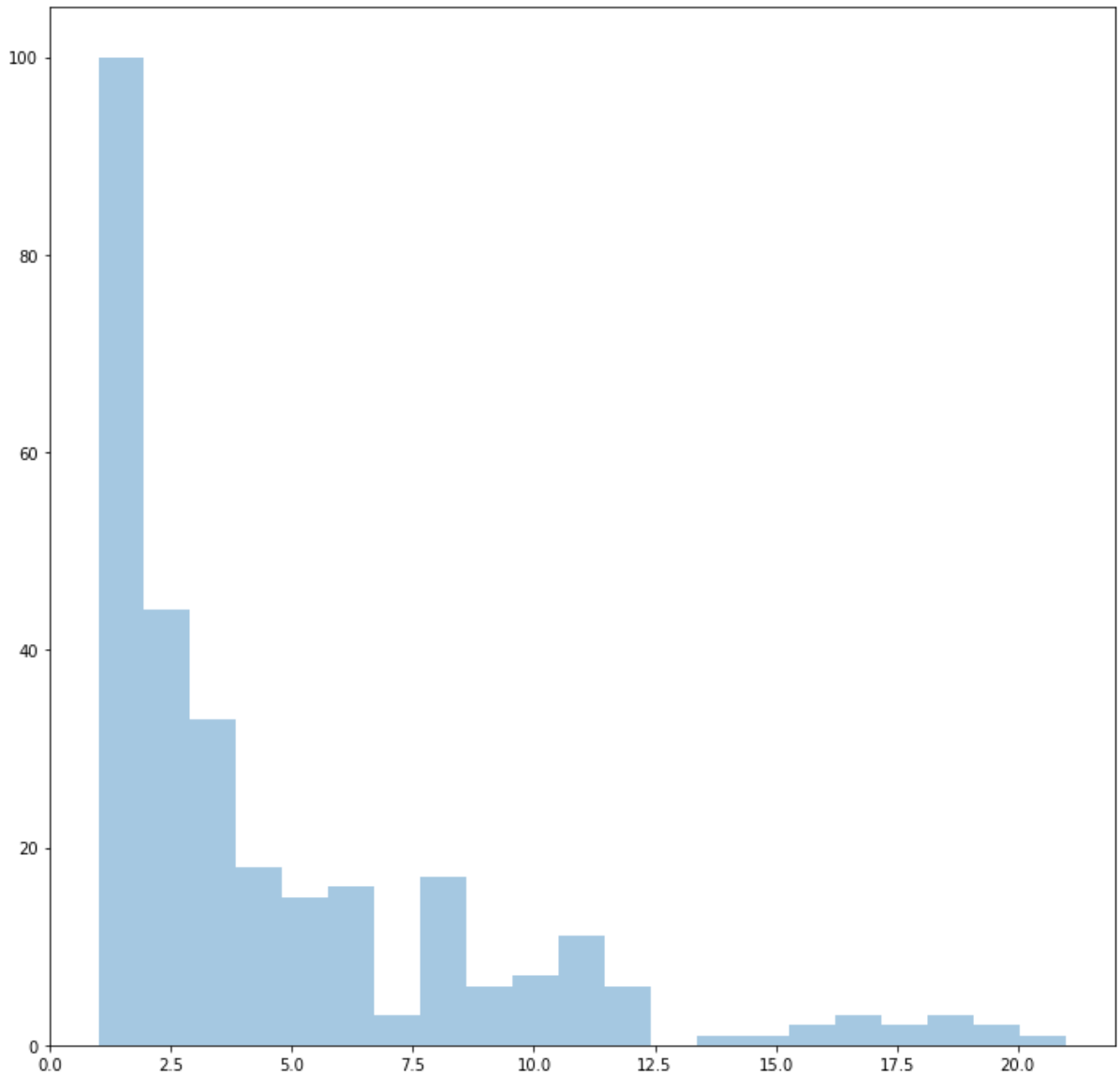
Imports

```
import csv
import itertools
import matplotlib.pyplot as plt
import matplotlib_venn as venn
import mpld3
import networkx as nx
import numpy as np
import scipy as sp
import seaborn as sns
import statistics

def edge_creator(sighting_group):
    sighting_group.sort()
    edges = [pair for pair in itertools.combinations(sighting_group, 2)]
    return edges
```

Importing Overall Network

```
with
open('C:/Users/Liam/Documents/DataMasters/MD2Networks/originalData/all_orig
.csv',
           'r') as fid:
    all_orig_text = fid.read()

lines = all_orig_text.split('\n')
lines = [line.split(",")[1:len(line.split(","))] for line in lines]
display(f"The overall network is built from {len(lines)} sightings")


overall_edge_list = [edge for line in lines for edge in edge_creator(line)]
overall_edge_list = list(set(overall_edge_list))


GrAll = nx.Graph()
for (x, y) in overall_edge_list:
    GrAll.add_node(x)
    GrAll.add_node(y)
    GrAll.add_edge(x, y)


display(f"GrAll has {GrAll.number_of_nodes()} nodes")
display(f"GrAll has {GrAll.number_of_edges()} edges")
'The overall network is built from 321 sightings'
'GrAll has 291 nodes'
'GrAll has 3182 edges'
dolphins = [node for node in GrAll.nodes]
#print(dolphins)
print(len(dolphins))
291
dolphins_seen = []

for line in lines:
    dolphins_seen = dolphins_seen + line
```

```
dolphin_count = [(dolphins_seen.count(dolphin), dolphin) for dolphin in
dolphins]
dolphin_count_dict = {dolphin:dolphins_seen.count(dolphin) for dolphin in
dolphins}

dolphin_count.sort(reverse = True)
max_count = max(dolphin_count)

dolphin_count_numbers, dolphin_count_dolpings = zip(*dolphin_count)

print(len(dolphin_count_numbers))


plt.figure(3,figsize=(12,12))
sns.distplot(dolphin_count_numbers, bins=max(dolphin_count_numbers),
hist=True, kde=False, rug=False)
291
<matplotlib.axes._subplots.AxesSubplot at 0x13987d81ac8>
```




```
max_count = max(dolphin_count_numbers)
mean_count = statistics.mean(dolphin_count_numbers)
median_count = statistics.median(dolphin_count_numbers)

display(f"GrAll has a maximum count of {max_count}")
```

```
display(f"GrAll has a mean count of {mean_count}")
display(f"GrAll has a median count of {median_count}")

# print(dolphin_count_numbers)
'GrAll has a maximum count of 21'
'GrAll has a mean count of 4.364261168384879'
'GrAll has a median count of 3'
```

Importing Social Network

```
with
open('C:/Users/Liam/Documents/DataMasters/MD2Networks/originalData/socialis
e_orig.csv',
          'r') as fid:
    social_orig_text = fid.read()

lines = social_orig_text.split('\n')
lines = [line.split(",")[1:len(line.split(","))] for line in lines]
display(f"The social network is built from {len(lines)} sightings")


social_edge_list = [edge for line in lines for edge in edge_creator(line)]
social_edge_list = list(set(social_edge_list))


GrSocial = nx.Graph()
for (x, y) in social_edge_list:
    GrSocial.add_node(x)
    GrSocial.add_node(y)
    GrSocial.add_edge(x, y)


display(f"GrSocial has {GrSocial.number_of_nodes()} nodes")
display(f"GrSocial has {GrSocial.number_of_edges()} edges")
'The social network is built from 40 sightings'
'GrSocial has 151 nodes'
'GrSocial has 1554 edges'
```

Importing Forage Network

```
with
open('C:/Users/Liam/Documents/DataMasters/MD2Networks/originalData/forage_o
rig.csv',
          'r') as fid:
    forage_orig_text = fid.read()

lines = forage_orig_text.split('\n')
lines = [line.split(",")[1:len(line.split(","))] for line in lines]
display(f"The forage network is built from {len(lines)} sightings")


forage_edge_list = [edge for line in lines for edge in edge_creator(line)]
forage_edge_list = list(set(forage_edge_list))


GrForage = nx.Graph()
for (x, y) in forage_edge_list:
    GrForage.add_node(x)
    GrForage.add_node(y)
    GrForage.add_edge(x, y)
```

```
display(f"GrForage has {GrForage.number_of_nodes()} nodes")
display(f"GrForage has {GrForage.number_of_edges()} edges")
'The forage network is built from 180 sightings'
'GrForage has 190 nodes'
'GrForage has 1134 edges'
```

Importing Travel Network

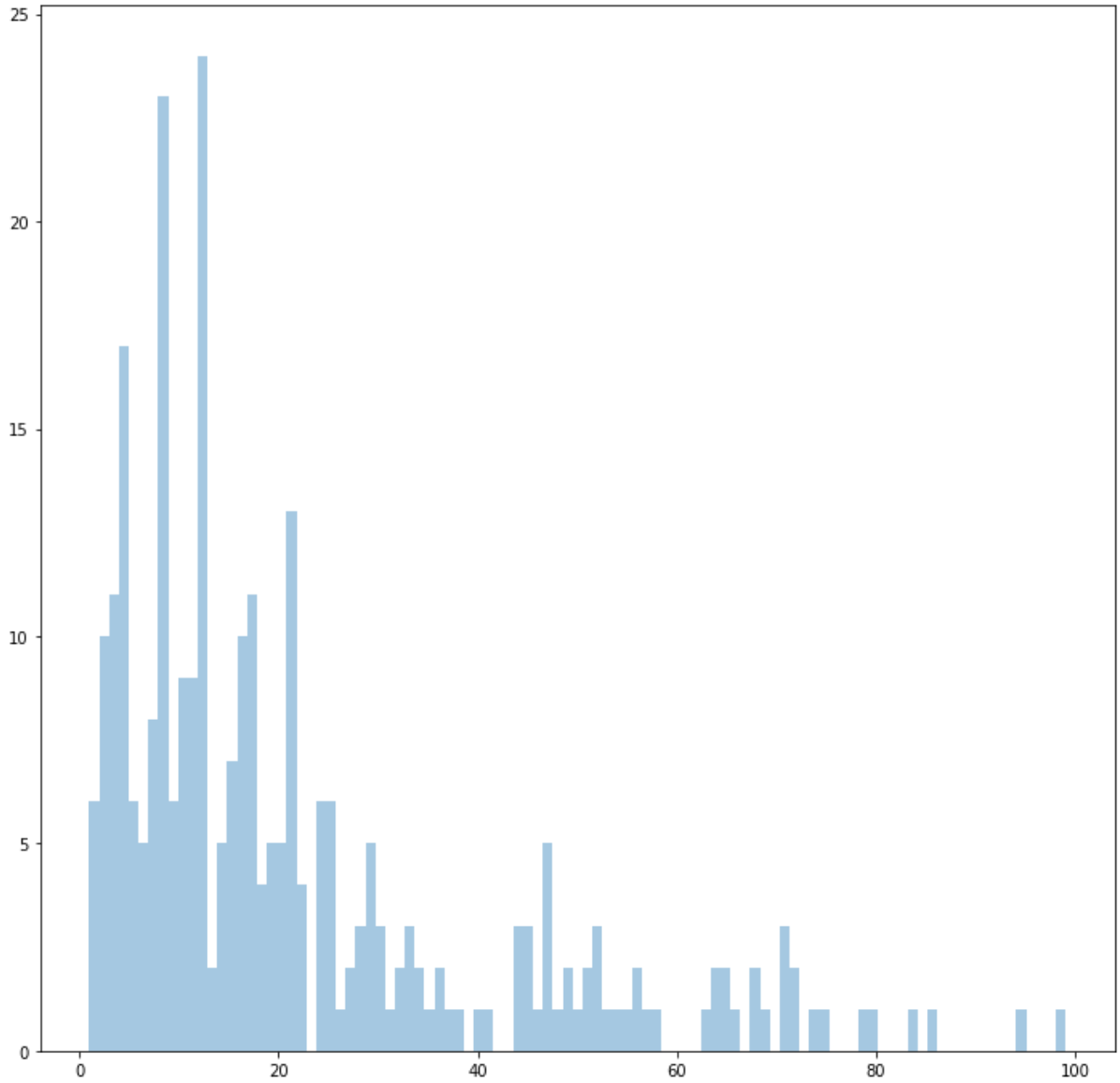
```
with
open('C:/Users/Liam/Documents/DataMasters/MD2Networks/originalData/travel_o
rig.csv',
          'r') as fid:
    travel_orig_text = fid.read()

lines = travel_orig_text.split('\n')
lines = [line.split(",")[1:len(line.split(","))] for line in lines]
display(f"The travel network is built from {len(lines)} sightings")


travel_edge_list = [edge for line in lines for edge in edge_creator(line)]
travel_edge_list = list(set(travel_edge_list))


GrTravel = nx.Graph()
for (x, y) in travel_edge_list:
    GrTravel.add_node(x)
    GrTravel.add_node(y)
    GrTravel.add_edge(x, y)


display(f"GrTravel has {GrTravel.number_of_nodes()} nodes")
display(f"GrTravel has {GrTravel.number_of_edges()} edges")
'The travel network is built from 103 sightings'
'GrTravel has 188 nodes'
'GrTravel has 1032 edges'
```

Basic Network Metrics

Overall

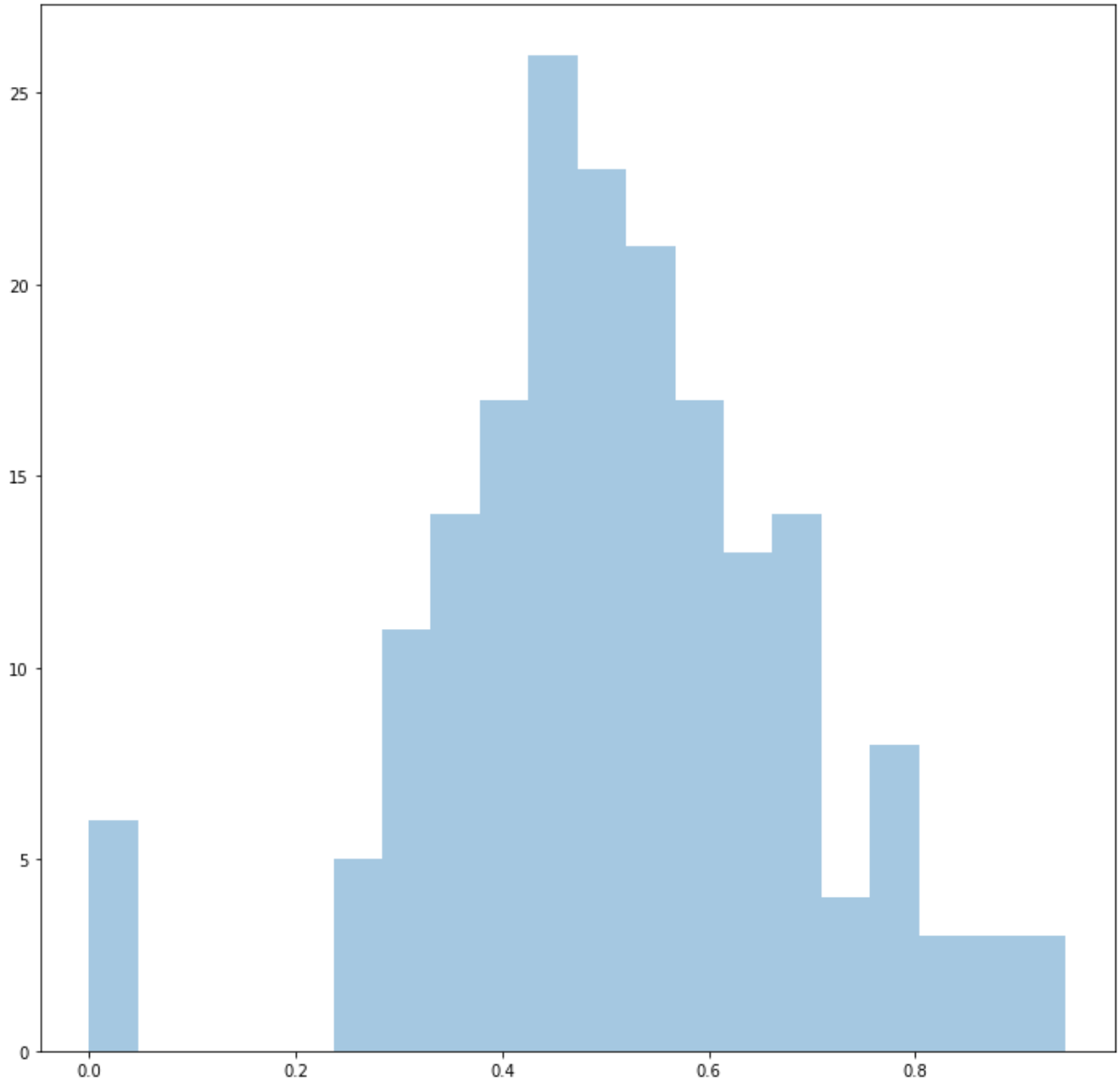
```
no_components_all = nx.number_connected_components(GrAll)
display(no_components_all)

components_all = [len(comp) for comp in nx.connected_components(GrAll)]
display(components_all)

nodes_all = [list(comp) for comp in nx.connected_components(GrAll)]
#print(nodes_all)

main_all = nodes_all[0]
splinter_all = [dolphin for x in range(1,no_components_all)for dolphin in
nodes_all[x] ]
print(splinter_all)

# diameter_all = nx.diameter(GrAll)
# print(diameter_all)
```

```
# all_connectivity = nx.node_connectivity(GrAll)
# display(all_connectivity)
3
[284, 5, 2]
['PTOS', 'U39O', 'NTOS', 'BEOS', 'KYOS', 'RRSR', 'WKSR']
GrAllMComp = nx.Graph()
for (x, y) in overall_edge_list:
    if x not in splinter_all:
        GrAllMComp.add_node(x)
        GrAllMComp.add_node(y)
        GrAllMComp.add_edge(x, y)


display(f"GrAllMComp has {GrAllMComp.number_of_nodes()} of GrAll's
{GrAll.number_of_nodes()} nodes")
display(f"GrAllMComp has {GrAllMComp.number_of_edges()} of GrAll's
{GrAll.number_of_edges()} edges")


diameter_all = nx.diameter(GrAllMComp)
display(f"GrAllMComp has a diameter of {diameter_all}")

radius_all = nx.radius(GrAllMComp)
display(f"GrAllMComp has a radius of {radius_all}")
"GrAllMComp has 284 of GrAll's 291 nodes"
"GrAllMComp has 3171 of GrAll's 3182 edges"
'GrAllMComp has a diameter of 6'
'GrAllMComp has a radius of 4'
degree_all = [GrAll.degree(dolphin) for dolphin in dolphins]
max_degree_all = max(degree_all)

mean_degree_all = statistics.mean(degree_all)
median_degree_all = statistics.median(degree_all)

display(f"GrAll has a maximum degree of {max_degree_all}")
display(f"GrAll has a mean degree of {mean_degree_all}")
display(f"GrAll has a median degree of {median_degree_all}")
# degree_all.sort()
# print(degree_all)
plt.figure(3,figsize=(12,12))
sns.distplot(degree_all, bins=max_degree_all, hist=True, kde=False,
rug=False)
'GrAll has a maximum degree of 99'
'GrAll has a mean degree of 21.869415807560138'
'GrAll has a median degree of 15'
<matplotlib.axes._subplots.AxesSubplot at 0x139882820b8>
```



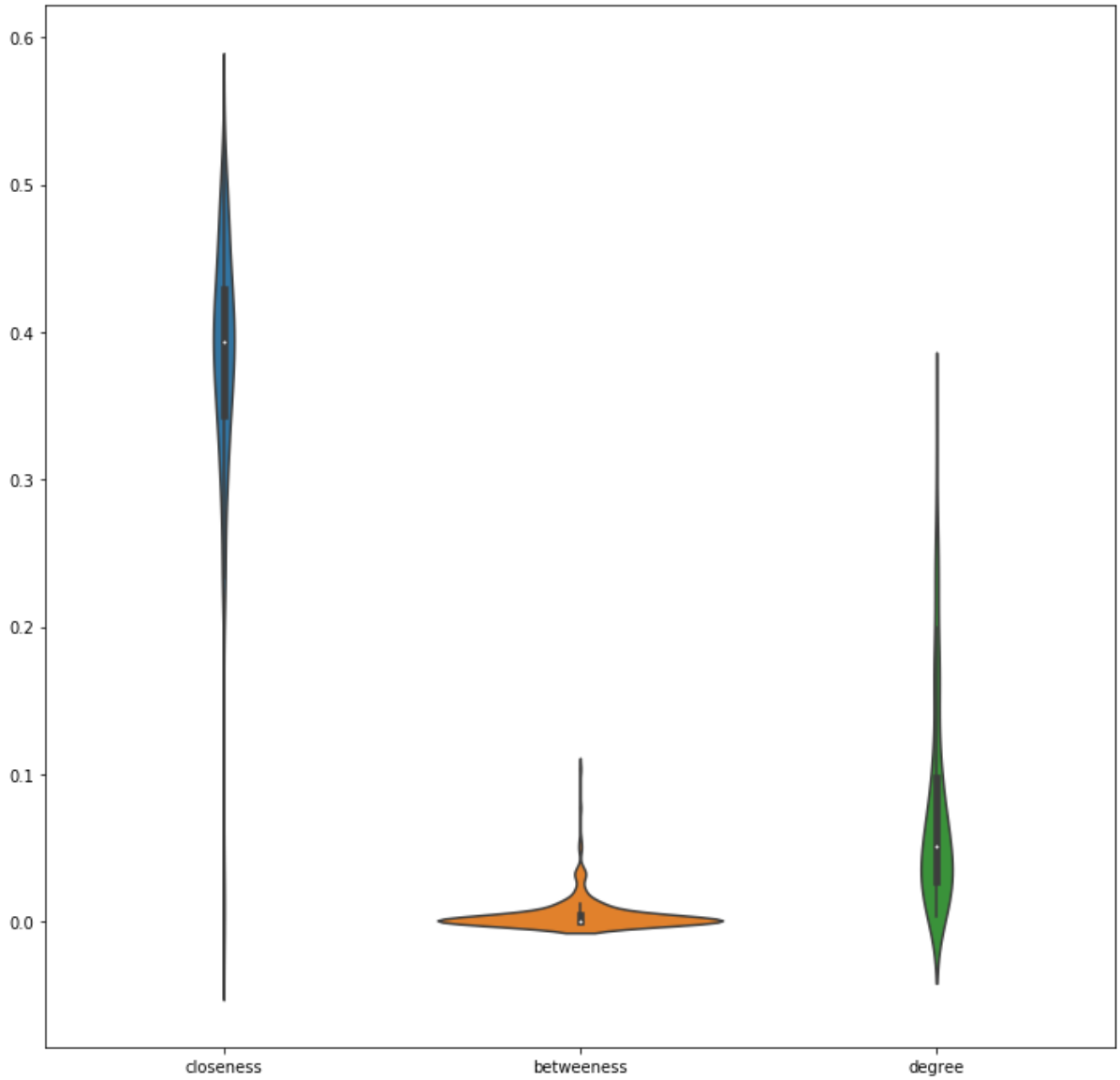

```
cluster_all = [nx.clustering(GrAll, nodes = dolphin) for dolphin in
dolphins]
# print(cluster_all)

i = 0
for x in cluster_all:
    if x == 1:
        i = i+1
print(i)
pct = 100*i/GrAll.number_of_nodes()

cluster_all_disunity = [coef for coef in cluster_all if coef < 1]
print(len(cluster_all_disunity))

display(f"{i} of GrAll's {GrAll.number_of_nodes()} nodes ({pct}%)have a
cluster coefficient of 1")

av_cluster_all = nx.average_clustering(GrAll)

display(f"The graph GrAll has an average cluster coefficient of
{av_cluster_all}")
```

```
plt.figure(3,figsize=(12,12))
sns.distplot(cluster_all_disunity, bins=20, hist=True, kde=False,
rug=False)

103
188
"103 of GrAll's 291 nodes (35.39518900343643%)have a cluster coefficient of
1"
'The graph GrAll has an average cluster coefficient of 0.6823326907114733'
<matplotlib.axes._subplots.AxesSubplot at 0x139884b54a8>
```



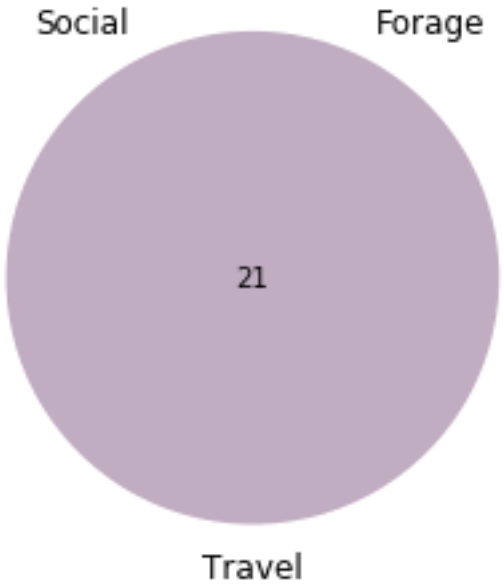

```
no_clique_all = nx.graph_number_of_cliques(GrAll)
clique_no_all = nx.graph_clique_number(GrAll)

display(f"The graph GrAll has {no_clique_all} cliques, the largest of which
contains {clique_no_all} dolphins")
'The graph GrAll has 942 cliques, the largest of which contains 31
dolphins'
tahi = nx.closeness_centrality(GrAll)
#display(tahi)
rua = nx.betweenness_centrality(GrAll, weight = None)
#display(rua)
toru = nx.degree_centrality(GrAll)
```

```
#display(toru)

list_closeness = [(tahi[dolphin], dolphin) for dolphin in dolphins]
list_closeness.sort(reverse = True)
list_closeness_dolphins = [list_closeness[x][1] for x in
range(0,len(dolphins))]
#display(list_closeness)
#display(list_closeness[0])
rank_closeness = {str(list_closeness[x][1]):x+1 for x in
range(0,len(list_closeness))}
#display(rank_closeness)

list_betweenness = [(rua[dolphin], dolphin) for dolphin in dolphins]
list_betweenness.sort(reverse = True)
list_betweenness_dolphins = [list_betweenness[x][1] for x in
range(0,len(dolphins))]
rank_betweenness = {str(list_betweenness[x][1]):x+1 for x in
range(0,len(list_betweenness))}

list_degree = [(toru[dolphin], dolphin) for dolphin in dolphins]
list_degree.sort(reverse = True)
list_degree_dolphins = [list_degree[x][1] for x in range(0,len(dolphins))]
rank_degree = {str(list_degree[x][1]):x+1 for x in
range(0,len(list_degree))}

list_overall = [(rank_closeness[str(dolphin)]
                 + rank_betweenness[str(dolphin)]
                 + rank_degree[str(dolphin)],
                  dolphin)
                for dolphin in dolphins]
list_overall.sort()
list_overall_dolphins = [list_overall[x][1] for x in
range(0,len(dolphins))]
rank_overall = {str(list_overall[x][1]):x+1 for x in
range(0,len(list_overall))}

# print(list_closeness)
# print(list_betweenness)
# print(list_degree)
centrality = {str(dolphin):{"closeness":tahi[dolphin],
                        "betweeness":rua[dolphin],
                        "degree":toru[dolphin],
                        "closeness_rank":rank_closeness[str(dolphin)],
                        "betweeness_rank":rank_betweenness[str(dolphin)],
                        "degree_rank":rank_degree[str(dolphin)],
                        "overall_rank":rank_overall[str(dolphin)]
                           }
              for dolphin in dolphins}
centrality_metrics = ('closeness', 'betweeness', 'degree')

plt.figure(3,figsize=(12,12))
sns.violinplot(x=[x for x in centrality_metrics for y in dolphins],
                   y=[centrality[str(y)][x] for x in centrality_metrics for
y in dolphins]
                  )
<matplotlib.axes._subplots.AxesSubplot at 0x13988871eb8>
```

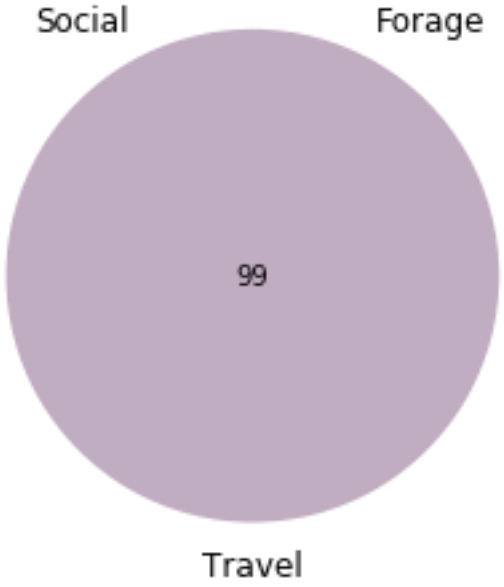



```
ffs_closeness = np.quantile([centrality[str(dolphin)]["closeness"] for
dolphin in dolphins],
                            (0,0.25, 0.5, 0.75, 1))
ffs_betweeness = np.quantile([centrality[str(dolphin)]["betweeness"] for
dolphin in dolphins],
                            (0,0.25, 0.5, 0.75, 1))
ffs_degree = np.quantile([centrality[str(dolphin)]["degree"] for dolphin in
dolphins],
                            (0,0.25, 0.5, 0.75, 1))

print(ffs_closeness)
print(ffs_betweeness)
print(ffs_degree)
[0.00344828 0.34392203 0.39340308 0.42916726 0.53314472]
[0.         0.         0.00085667 0.00523156 0.10392713]
[0.00344828 0.02758621 0.05172414 0.09827586 0.34137931]

central_dolphins = list_overall_dolphins[0:5]
central_dolphins.append(list_closeness_dolphins[0])
central_dolphins.append(list_betweenness_dolphins[0])
central_dolphins.append(list_degree_dolphins[0])

central_dolphins = list(set(central_dolphins))
```

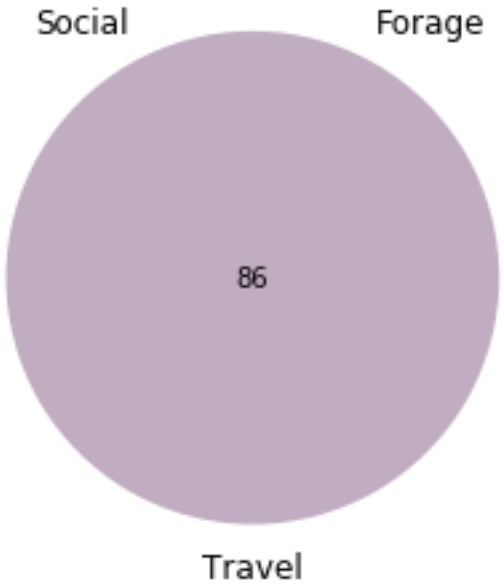
```
central_dolphins.sort()
display(central_dolphins)
['CRSC', 'KKNK', 'LKWB', 'MHWB', 'OLWB', 'SOCT']
# G = nx.GrAll()   # or DiGraph, MultiGraph, MultiDiGraph, etc

neighbours = list(GrAll.neighbors('CRSC'))
# display(neighbours)
networks = ['social', 'forage', 'travel']

# for network in networks:
#     print(network)

# for dolphin in central_dolphins:
#     print(dolphin)
central_dolphin_count = [(dolphins_seen.count(dolphin), dolphin) for
dolphin in central_dolphins]
print(central_dolphin_count)
[(20, 'CRSC'), (3, 'KKNK'), (11, 'LKWB'), (18, 'MHWB'), (20, 'OLWB'), (18,
'SOCT')]


networks = ['social', 'forage', 'travel']

central_neighborhoods ={}
# central_neighborhoods['CRSC'] = {}
# central_neighborhoods['CRSC']['social'] = list(GrAll.neighbors('CRSC'))

for dolphin in central_dolphins:
    central_neighborhoods[dolphin] = {}
    for network in networks:
        central_neighborhoods[dolphin][network] =
list(GrAll.neighbors(dolphin))


# print(central_neighborhoods)

# print(central_neighborhoods['KKNK']['social'])
# print(central_neighborhoods['KKNK']['forage'])
# print(central_neighborhoods['KKNK']['travel'])
from matplotlib_venn import venn3, venn3_circles
v = venn3([set(central_neighborhoods['KKNK']['social']),
           set(central_neighborhoods['KKNK']['forage']),
          set(central_neighborhoods['KKNK']['travel'])],
      set_labels = ('Social', 'Forage', 'Travel'))
```



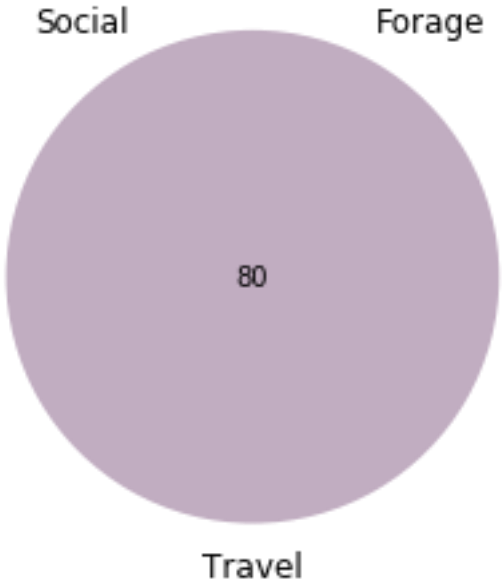

```
v = venn3([set(central_neighborhoods['CRSC']['social']),
           set(central_neighborhoods['CRSC']['forage']),
          set(central_neighborhoods['CRSC']['travel'])],
      set_labels = ('Social', 'Forage', 'Travel'))
```




```
v = venn3([set(central_neighborhoods['MHWB']['social']),
           set(central_neighborhoods['MHWB']['forage']),
          set(central_neighborhoods['MHWB']['travel'])],
      set_labels = ('Social', 'Forage', 'Travel'))
```

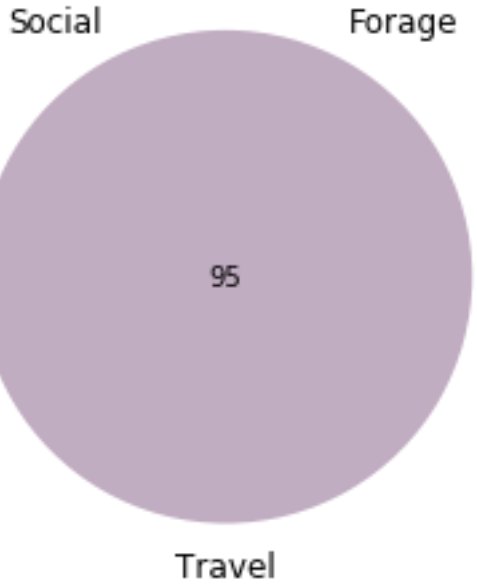



```
central_edges = [list(GrAll.edges(dolphin)) for dolphin in
central_dolphins]
# central_edges = nx.GrAll.edges('IHWB')
#print(central_edges)
all_central_edges =[]

for edgelist in central_edges:
    all_central_edges = all_central_edges + edgelist

#print(all_central_edges)
#LKWB
v = venn3([set(central_neighborhoods['LKWB']['social']),
           set(central_neighborhoods['LKWB']['forage']),
          set(central_neighborhoods['LKWB']['travel'])],
      set_labels = ('Social', 'Forage', 'Travel'))
```

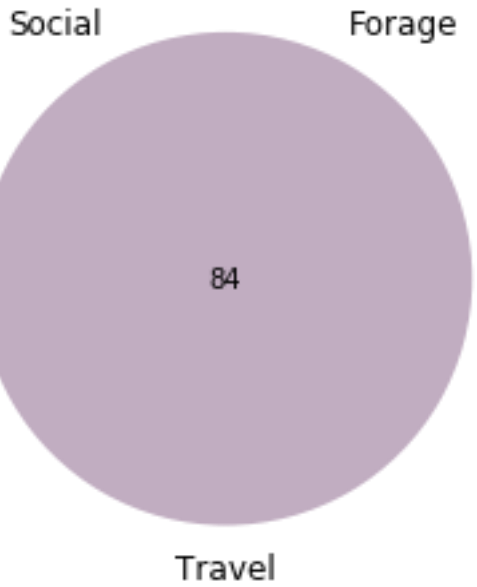



```
#OLWB
v = venn3([set(central_neighborhoods['OLWB']['social']),
           set(central_neighborhoods['OLWB']['forage']),
          set(central_neighborhoods['OLWB']['travel'])],
```

```
      set_labels = ('Social', 'Forage', 'Travel'))
```



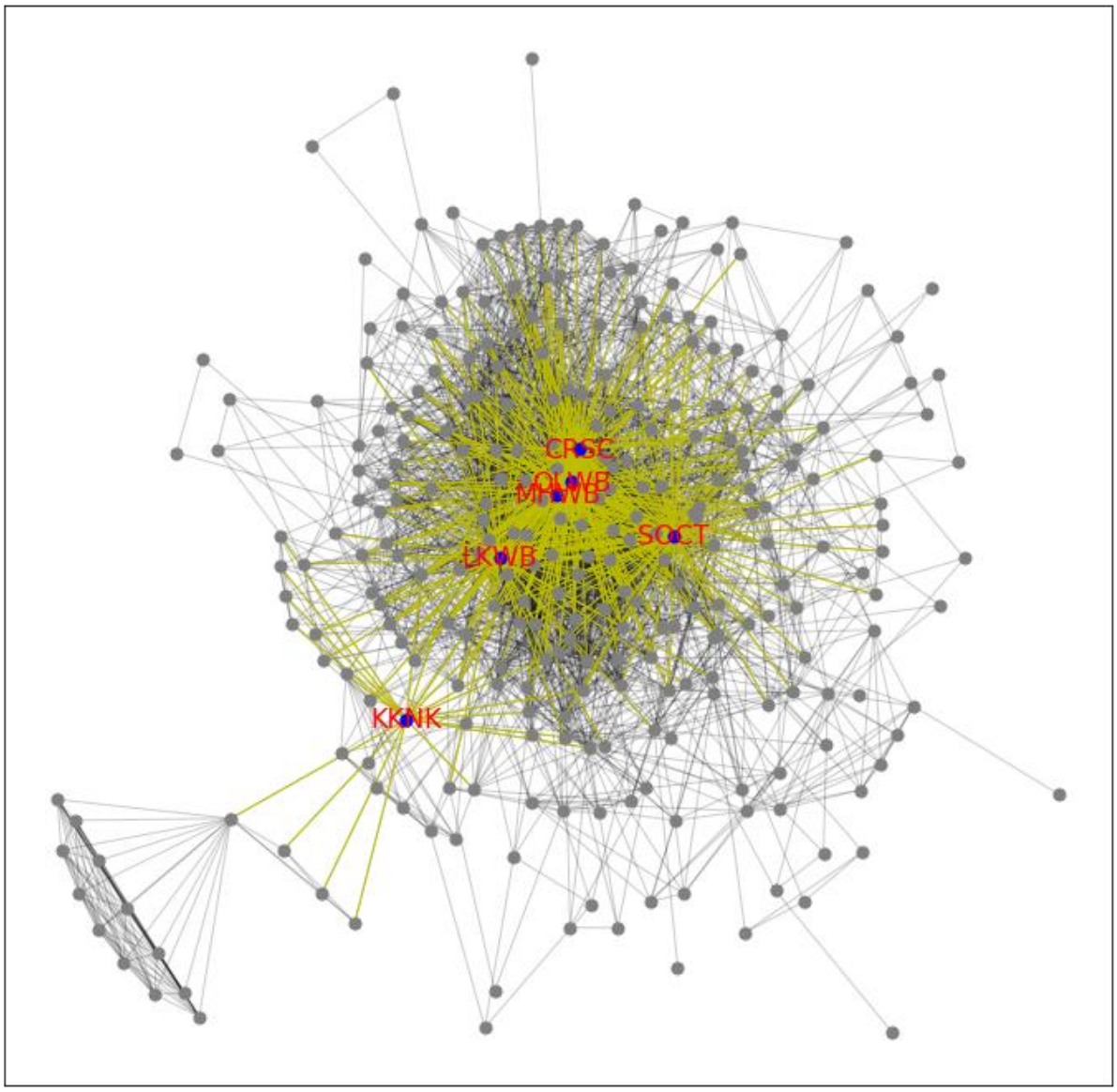

```
#SOCT
v = venn3([set(central_neighborhoods['SOCT']['social']),
           set(central_neighborhoods['SOCT']['forage']),
          set(central_neighborhoods['SOCT']['travel'])],
      set_labels = ('Social', 'Forage', 'Travel'))
```




```
for dolphin in central_dolphins:
    display(dolphin, centrality[str(dolphin)])
'CRSC'
{'closeness': 0.5220585359494165,
 'betweeness': 0.05607821479813801,
 'degree': 0.3413793103448276,
 'closeness_rank': 2,
 'betweeness_rank': 3,
 'degree_rank': 1,
 'overall_rank': 1}
'KKNK'
{'closeness': 0.42034850154831255,
 'betweeness': 0.10392713143437239,
```

```
 'degree': 0.07241379310344828,
 'closeness_rank': 88,
 'betweeness_rank': 1,
 'degree_rank': 103,
 'overall_rank': 58}
'LKWB'
{'closeness': 0.5048792788249385,
 'betweeness': 0.033883586384414056,
 'degree': 0.27586206896551724,
 'closeness_rank': 6,
 'betweeness_rank': 10,
 'degree_rank': 5,
 'overall_rank': 4}
'MHWB'
{'closeness': 0.5171703474105643,
 'betweeness': 0.022404208799274898,
 'degree': 0.296551724137931,
 'closeness_rank': 3,
 'betweeness_rank': 18,
 'degree_rank': 3,
 'overall_rank': 5}
'OLWB'
{'closeness': 0.5331447210757555,
 'betweeness': 0.03633684284167598,
 'degree': 0.3275862068965517,
 'closeness_rank': 1,
 'betweeness_rank': 6,
 'degree_rank': 2,
 'overall_rank': 2}
'SOCT'
{'closeness': 0.5142811275926283,
 'betweeness': 0.05114276681432285,
 'degree': 0.2896551724137931,
 'closeness_rank': 4,
 'betweeness_rank': 4,
 'degree_rank': 4,
 'overall_rank': 3}
pos = nx.kamada_kawai_layout(GrAll)
plt.figure(3,figsize=(12,12))
nx.draw_networkx_nodes(GrAll, pos, fixed = True, with_labels=False,
node_size = 50,
                       node_color = '#808080', font_size = 6, alpha = 1)
nx.draw_networkx_nodes(GrAll, pos, fixed = True, with_labels=True,
node_size = 50,
                       node_color = 'b', font_size = 10, alpha = 1,
                       nodelist=central_dolphins)
nx.draw_networkx_edges(GrAll, pos, fixed = True, alpha = 0.2)
nx.draw_networkx_edges(GrAll, pos, edgelist = all_central_edges, fixed =
True, edge_color = 'y', alpha = 1)

labels = {}
for dolphin in dolphins:
    if dolphin in central_dolphins:
        #set the node name as the key and the label as its value
        labels[dolphin] = dolphin

# fig, ax = plt.subplots(subplot_kw=dict(facecolor='#EEEEEE'))
# scatter = nx.draw_networkx_nodes(GrAll, pos)

# labels_all = GrAll.nodes()
# tooltip = mpld3.plugins.PointLabelTooltip(scatter, labels=labels_all)
```

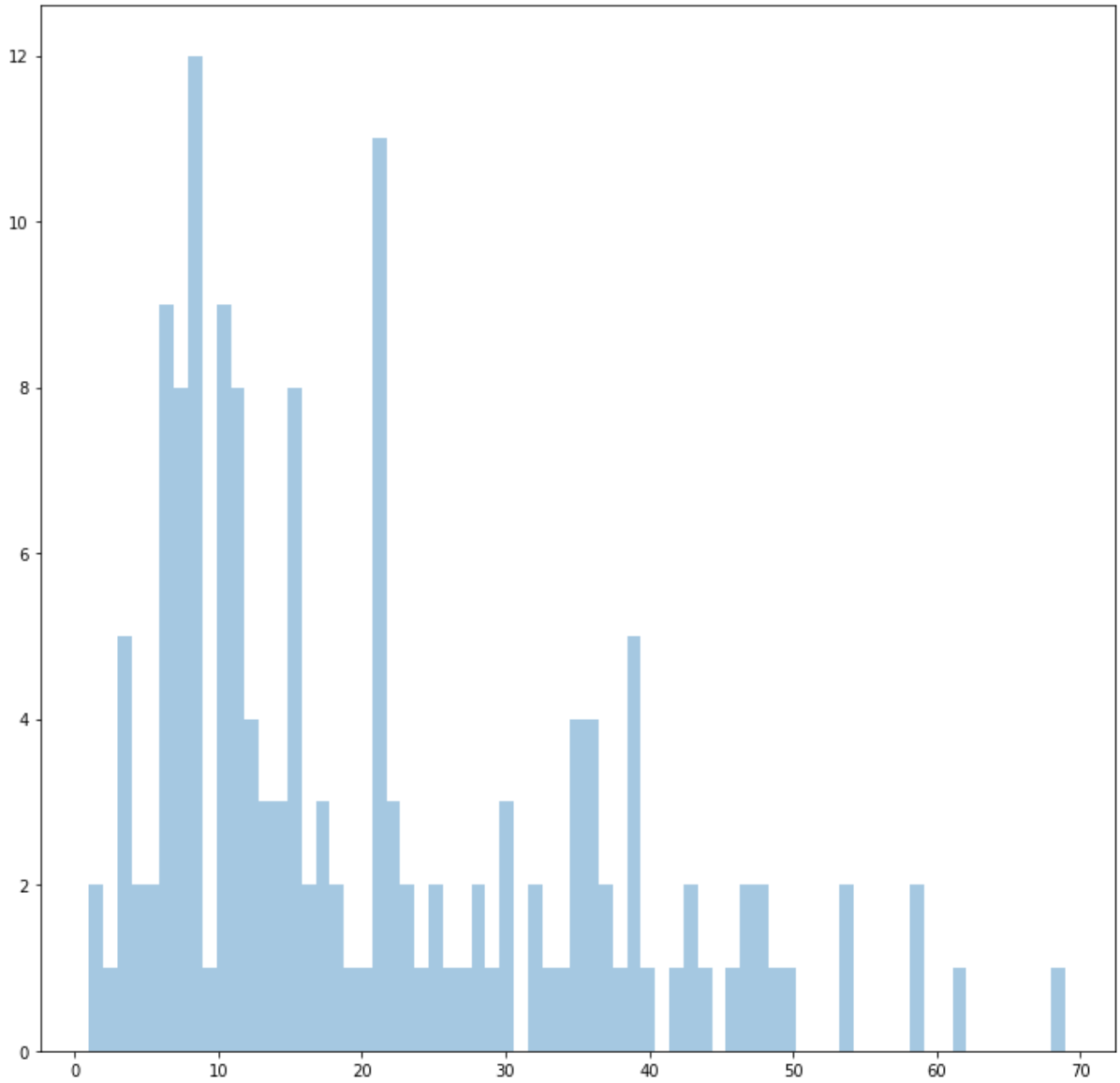
```
# mpld3.plugins.connect(fig, tooltip)

# mpld3.show()

nx.draw_networkx_labels(GrAll, pos, labels, font_size=16,font_color='r')
plt.show()
C:\Users\Liam\Anaconda3\lib\site-packages\networkx\drawing\nx_pylab.py:579:
MatplotlibDeprecationWarning:
The iterable function was deprecated in Matplotlib 3.1 and will be removed
in 3.3. Use np.iterable instead.
  if not cb.iterable(width):
```



Social

```
no_components_social = nx.number_connected_components(GrSocial)
display(no_components_social)

components_social = [len(comp) for comp in
nx.connected_components(GrSocial)]
display(components_social)

nodes_social = [list(comp) for comp in nx.connected_components(GrSocial)]
```

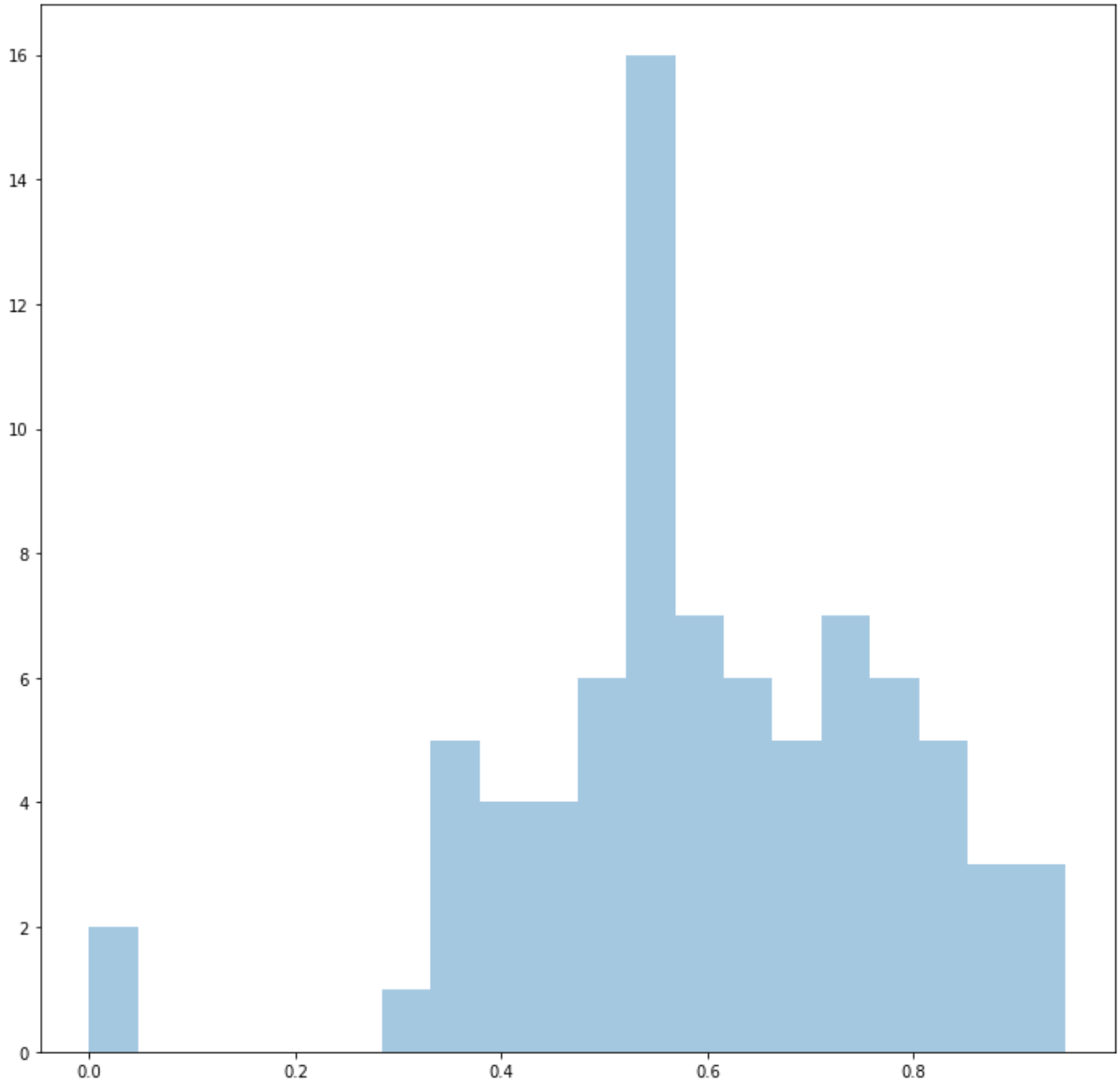
```
#print(nodes_all)

main_social = nodes_social[0]
splinter_social = [dolphin for x in range(1,no_components_social)for
dolphin in nodes_social[x] ]
print(splinter_social)

# diameter_all = nx.diameter(GrAll)
# print(diameter_all)

# all_connectivity = nx.node_connectivity(GrAll)
# display(all_connectivity)
2
[149, 2]
['RRSR', 'WKSR']
GrSocialMComp = nx.Graph()
for (x, y) in social_edge_list:
    if x not in splinter_social:
        GrSocialMComp.add_node(x)
        GrSocialMComp.add_node(y)
        GrSocialMComp.add_edge(x, y)


display(f"GrSocialMComp has {GrSocialMComp.number_of_nodes()} of GrSocial's
{GrSocial.number_of_nodes()} nodes")
display(f"GrSocialMComp has {GrSocialMComp.number_of_edges()} of GrSocial's
{GrSocial.number_of_edges()} edges")


diameter_social = nx.diameter(GrSocialMComp)
display(f"GrSocialMComp has a diameter of {diameter_social}")

radius_social = nx.radius(GrSocialMComp)
display(f"GrSocialMComp has a radius of {radius_social}")
"GrSocialMComp has 149 of GrSocial's 151 nodes"
"GrSocialMComp has 1553 of GrSocial's 1554 edges"
'GrSocialMComp has a diameter of 4'
'GrSocialMComp has a radius of 2'
degree_social = [GrSocial.degree(dolphin) for dolphin in dolphins if
dolphin in GrSocial]
max_degree_social = max(degree_social)

mean_degree_social = statistics.mean(degree_social)
median_degree_social = statistics.median(degree_social)

display(f"GrSocial has a maximum degree of {max_degree_social}")
display(f"GrSocial has a mean degree of {mean_degree_social}")
display(f"GrSocial has a median degree of {median_degree_social}")

plt.figure(3,figsize=(12,12))
sns.distplot(degree_social, bins=max_degree_social, hist=True, kde=False,
rug=False)
'GrSocial has a maximum degree of 69'
'GrSocial has a mean degree of 20.582781456953644'
'GrSocial has a median degree of 15'
<matplotlib.axes._subplots.AxesSubplot at 0x13988ba7978>
```




```
cluster_social = [nx.clustering(GrSocial, nodes = dolphin) for dolphin in
dolphins if dolphin in GrSocial]
# print(cluster_social)

i = 0
for x in cluster_social:
    if x == 1:
        i = i+1
print(i)
pct = 100*i/GrSocial.number_of_nodes()

cluster_social_disunity = [coef for coef in cluster_social if coef < 1]

display(f"{i} of GrSocial's {GrSocial.number_of_nodes()} nodes ({pct}%)have
a cluster coefficient of 1")

av_cluster_social = nx.average_clustering(GrSocial)

display(f"The graph GrSocial has an average cluster coefficient of
{av_cluster_social}")

plt.figure(3,figsize=(12,12))
```

```
sns.distplot(cluster_social_disunity, bins=20, hist=True, kde=False,
rug=False)
```

```
71
"71 of GrSocial's 151 nodes (47.019867549668874%)have a cluster coefficient
of 1"
'The graph GrSocial has an average cluster coefficient of
0.7882579347214447'
<matplotlib.axes._subplots.AxesSubplot at 0x139892225f8>
```

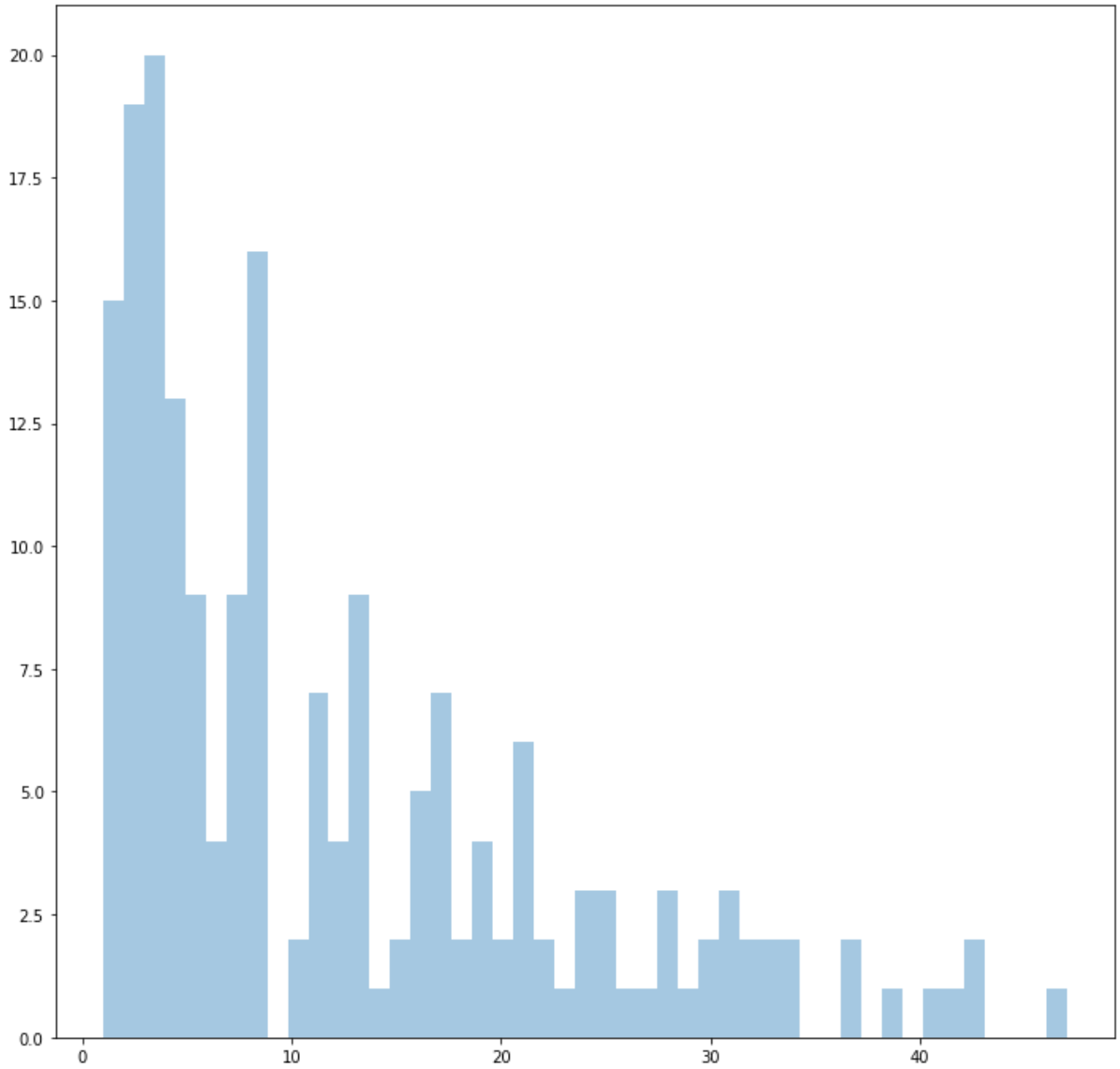



```
no_clique_social = nx.graph_number_of_cliques(GrSocial)
clique_no_social = nx.graph_clique_number(GrSocial)

display(f"The graph GrSocial has {no_clique_social} cliques, the largest of
which contains {clique_no_social} dolphins")
'The graph GrSocial has 116 cliques, the largest of which contains 31
dolphins'
```

Forage

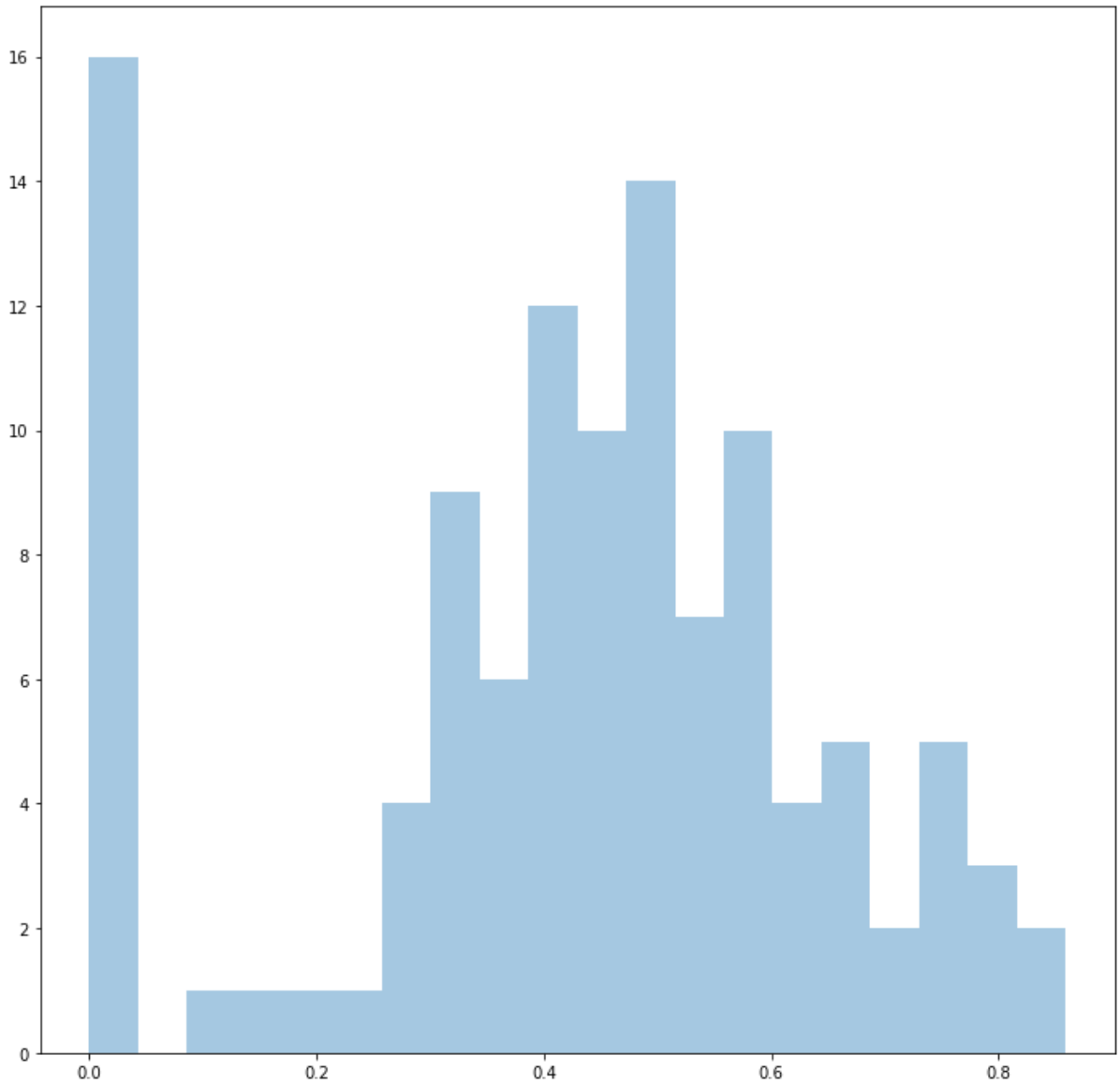
```
no_components_forage = nx.number_connected_components(GrForage)
display(no_components_forage)
```

```
components_forage = [len(comp) for comp in
nx.connected_components(GrForage)]
display(components_forage)

nodes_forage = [list(comp) for comp in nx.connected_components(GrForage)]
#print(nodes_all)

main_forage = nodes_forage[0]
splinter_forage = [dolphin for x in range(1,no_components_forage)for
dolphin in nodes_forage[x] ]
# print(splinter_forage)

# diameter_all = nx.diameter(GrAll)
# print(diameter_all)

# all_connectivity = nx.node_connectivity(GrAll)
# display(all_connectivity)
6
[176, 4, 2, 3, 3, 2]
GrForageMComp = nx.Graph()
for (x, y) in forage_edge_list:
    if x not in splinter_forage:
        GrForageMComp.add_node(x)
        GrForageMComp.add_node(y)
        GrForageMComp.add_edge(x, y)


display(f"GrForageMComp has {GrForageMComp.number_of_nodes()} of GrForage's
{GrForage.number_of_nodes()} nodes")
display(f"GrForageMComp has {GrForageMComp.number_of_edges()} of GrForage's
{GrForage.number_of_edges()} edges")


diameter_forage = nx.diameter(GrForageMComp)
display(f"GrForageMComp has a diameter of {diameter_forage}")

radius_forage = nx.radius(GrForageMComp)
display(f"GrForageMComp has a radius of {radius_forage}")
"GrForageMComp has 176 of GrForage's 190 nodes"
"GrForageMComp has 1121 of GrForage's 1134 edges"
'GrForageMComp has a diameter of 7'
'GrForageMComp has a radius of 4'
degree_forage = [GrForage.degree(dolphin) for dolphin in dolphins if
dolphin in GrForage]
max_degree_forage = max(degree_forage)

mean_degree_forage = statistics.mean(degree_forage)
median_degree_forage = statistics.median(degree_forage)

display(f"GrForage has a maximum degree of {max_degree_forage}")
display(f"GrForage has a mean degree of {mean_degree_forage}")
display(f"GrForage has a median degree of {median_degree_forage}")

plt.figure(3,figsize=(12,12))
sns.distplot(degree_forage, bins=max_degree_forage, hist=True, kde=False,
rug=False)
'GrForage has a maximum degree of 47'
'GrForage has a mean degree of 11.936842105263159'
'GrForage has a median degree of 8.0'
<matplotlib.axes._subplots.AxesSubplot at 0x1398a4f1518>
```




```
cluster_forage = [nx.clustering(GrForage, nodes = dolphin) for dolphin in
dolphins if dolphin in GrForage]
# print(cluster_forage)

i = 0
for x in cluster_forage:
    if x == 1:
        i = i+1
print(i)
pct = 100*i/GrForage.number_of_nodes()

cluster_forage_disunity = [coef for coef in cluster_forage if coef < 1]

display(f"{i} of GrForage's {GrForage.number_of_nodes()} nodes ({pct}%)have
a cluster coefficient of 1")

av_cluster_forage = nx.average_clustering(GrForage)

display(f"The graph GrForage has an average cluster coefficient of
{av_cluster_forage}")

plt.figure(3,figsize=(12,12))
```

```
sns.distplot(cluster_forage_disunity, bins=20, hist=True, kde=False,
rug=False)
77
"77 of GrForage's 190 nodes (40.526315789473685%)have a cluster coefficient
of 1"
'The graph GrForage has an average cluster coefficient of
0.6587530358006516'
<matplotlib.axes._subplots.AxesSubplot at 0x1398a9ba860>
```

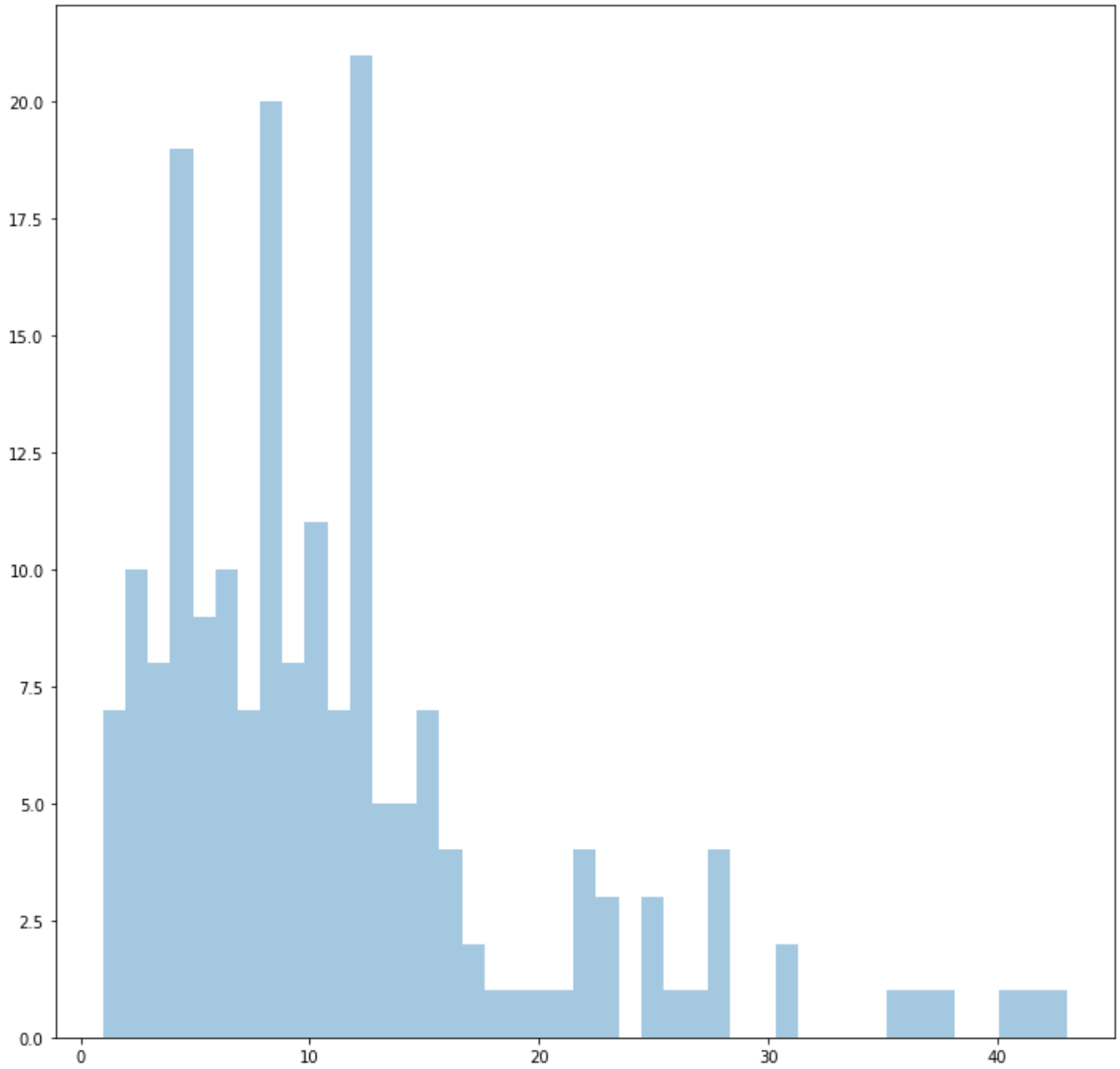



```
no_clique_forage = nx.graph_number_of_cliques(GrForage)
clique_no_forage = nx.graph_clique_number(GrForage)

display(f"The graph GrForage has {no_clique_forage} cliques, the largest of
which contains {clique_no_forage} dolphins")
'The graph GrForage has 136 cliques, the largest of which contains 18
dolphins'
```

Travel

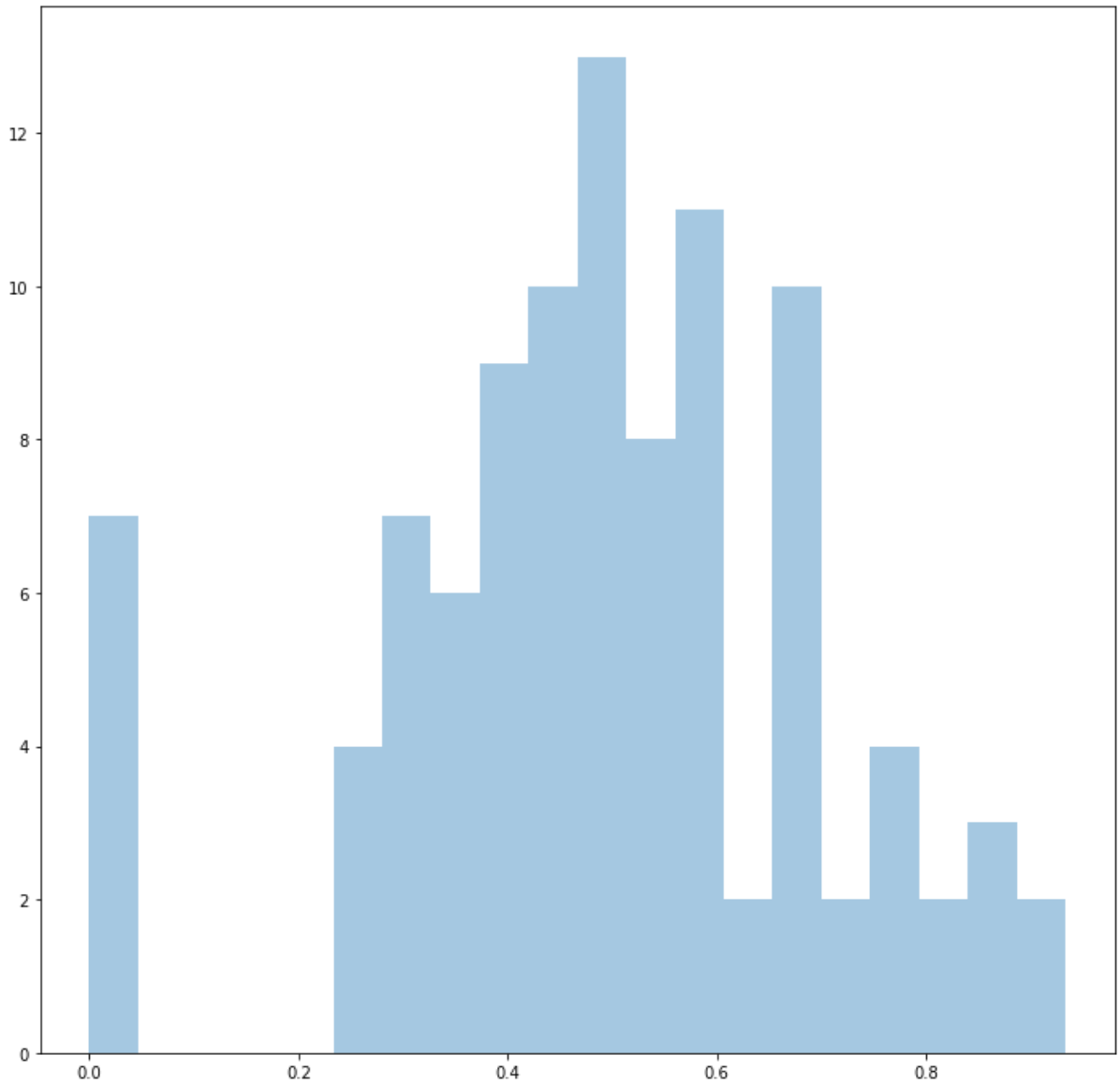
```
no_components_travel = nx.number_connected_components(GrTravel)
display(no_components_travel)
```

```
components_travel = [len(comp) for comp in
nx.connected_components(GrTravel)]
display(components_travel)

nodes_travel = [list(comp) for comp in nx.connected_components(GrTravel)]
#print(nodes_all)

main_travel = nodes_travel[0]
splinter_travel = [dolphin for x in range(1,no_components_travel)for
dolphin in nodes_travel[x] ]
# print(splinter_travel)
6
[17, 157, 4, 3, 5, 2]
GrTravelMComp = nx.Graph()
for (x, y) in travel_edge_list:
    if x not in splinter_travel:
        GrTravelMComp.add_node(x)
        GrTravelMComp.add_node(y)
        GrTravelMComp.add_edge(x, y)


display(f"GrTravelMComp has {GrTravelMComp.number_of_nodes()} of GrTravel's
{GrTravel.number_of_nodes()} nodes")
display(f"GrTravelMComp has {GrTravelMComp.number_of_edges()} of GrTravel's
{GrTravel.number_of_edges()} edges")


diameter_travel = nx.diameter(GrTravelMComp)
display(f"GrTravelMComp has a diameter of {diameter_travel}")

radius_travel = nx.radius(GrTravelMComp)
display(f"GrTravelMComp has a radius of {radius_travel}")
"GrTravelMComp has 17 of GrTravel's 188 nodes"
"GrTravelMComp has 88 of GrTravel's 1032 edges"
'GrTravelMComp has a diameter of 2'
'GrTravelMComp has a radius of 1'
degree_travel = [GrTravel.degree(dolphin) for dolphin in dolphins if
dolphin in GrTravel]
max_degree_travel = max(degree_travel)

mean_degree_travel = statistics.mean(degree_travel)
median_degree_travel = statistics.median(degree_travel)

display(f"GrTravel has a maximum degree of {max_degree_travel}")
display(f"GrTravel has a mean degree of {mean_degree_travel}")
display(f"GrTravel has a median degree of {median_degree_travel}")

plt.figure(3,figsize=(12,12))
sns.distplot(degree_travel, bins=max_degree_travel, hist=True, kde=False,
rug=False)
'GrTravel has a maximum degree of 43'
'GrTravel has a mean degree of 10.97872340425532'
'GrTravel has a median degree of 9.0'
<matplotlib.axes._subplots.AxesSubplot at 0x13988b02fd0>
```




```
cluster_travel = [nx.clustering(GrTravel, nodes = dolphin) for dolphin in
dolphins if dolphin in GrTravel]
# print(cluster_travel)

i = 0
for x in cluster_travel:
    if x == 1:
        i = i+1
print(i)
pct = 100*i/GrTravel.number_of_nodes()

cluster_travel_disunity = [coef for coef in cluster_travel if coef < 1]

display(f"{i} of GrTravel's {GrTravel.number_of_nodes()} nodes ({pct}%)have
a cluster coefficient of 1")

av_cluster_travel = nx.average_clustering(GrTravel)

display(f"The graph GrTravel has an average cluster coefficient of
{av_cluster_travel}")

plt.figure(3,figsize=(12,12))
```

```
sns.distplot(cluster_travel_disunity, bins=20, hist=True, kde=False,
rug=False)
88
"88 of GrTravel's 188 nodes (46.808510638297875%)have a cluster coefficient
of 1"
'The graph GrTravel has an average cluster coefficient of
0.727745909291052'
<matplotlib.axes._subplots.AxesSubplot at 0x1398aa53438>
```

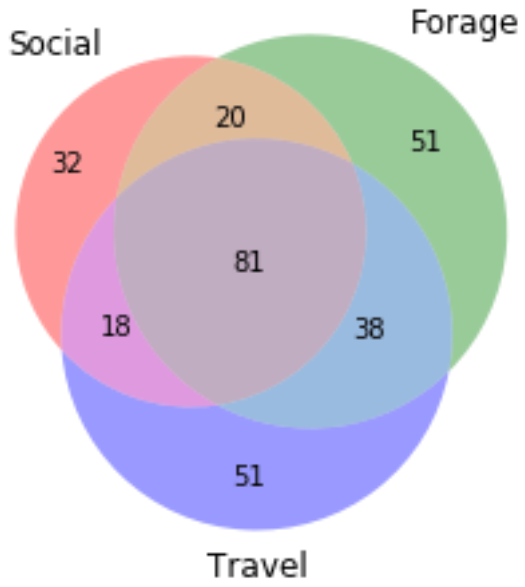



```
no_clique_travel = nx.graph_number_of_cliques(GrTravel)
clique_no_travel = nx.graph_clique_number(GrTravel)

display(f"The graph GrTravel has {no_clique_travel} cliques, the largest of
which contains {clique_no_travel} dolphins")
'The graph GrTravel has 131 cliques, the largest of which contains 16
dolphins'
```

Considering Network Overlap

```
dolphins_social = [node for node in GrSocial.nodes]
dolphins_forage = [node for node in GrForage.nodes]
dolphins_travel = [node for node in GrTravel.nodes]
```

```
print(type(dolphins_social))
<class 'list'>
dolphins_set = set(dolphins)
dolphins_social_set = set(dolphins_social)
dolphins_forage_set = set(dolphins_forage)
dolphins_travel_set = set(dolphins_travel)

print(len(dolphins_set))
print(len(dolphins_social_set))
print(len(dolphins_forage_set))
print(len(dolphins_travel_set))
291
151
190
188
dolphins_soc_for = dolphins_social_set.intersection(dolphins_forage_set)
dolphins_for_tra = dolphins_forage_set.intersection(dolphins_travel_set)
dolphins_tra_soc = dolphins_travel_set.intersection(dolphins_social_set)

dolphins_tra_soc_for = dolphins_tra_soc.intersection(dolphins_forage_set)

print(len(dolphins_soc_for))
print(len(dolphins_for_tra))
print(len(dolphins_tra_soc))
print(len(dolphins_tra_soc_for))
101
119
99
81
from matplotlib_venn import venn3, venn3_circles
v = venn3([dolphins_social_set, dolphins_forage_set, dolphins_travel_set],
      set_labels = ('Social', 'Forage', 'Travel'))
```




```
check = 32 + 20 + 51 + 18 + 81 + 38 + 51
print(check)

dolphins_lost = [dolphin for dolphin in dolphins
                 if dolphin not in dolphins_social
                 and dolphin not in dolphins_forage
                and dolphin not in dolphins_travel]
```

```
print(len(dolphins_lost))
291
0
```