

17

Analysis of Graphs

图的结构

度分析、图距离、中心性、社区



游迹国家全域，您可以走御道，条条大路任由百姓选择；但是，几何学习只有一条成功之路。

For traveling over the country, there are royal road and roads for common citizens, but in geometry there is one road for all.

—— 梅内克谟斯 (Menaechmus) | 古希腊数学家 | 380 ~ 320 BC



```

networkx.algorithms.community centrality.girvan_newman() Girvan-Newman 算法划分社区
networkx.betweenness Centrality() 计算介数中心性
networkx.center() 找出图的中心节点，即离心率等于图半径的所有节点
networkx.closeness Centrality() 计算紧密中心性
networkx.connected_components() 计算图中连通分量
networkx.degree Centrality() 计算度中心性
networkx.diameter() 计算图的直径，即图中所有节点离心率的最大值
networkx.eccentricity() 计算图中每个节点的离心率，即该节点图距离的最大值
networkx.eigenvector Centrality() 计算特征向量中心性
networkx.periphery() 找出图的边缘节点，即离心率等于图直径的所有节点
networkx.radius() 计算图的半径，即图中所有节点的离心率的最小值
networkx.shortest_path() 寻找两个节点之间的最短路径
networkx.shortest_path_length() 计算在图中两个节点之间的最短路径的长度
numpy.tril() 生成一个数组的下三角矩阵，其余部分填充为零
numpy.tril_indices() 返回一个数组下三角矩阵的索引
numpy.unique() 找出数组中所有唯一值并返回已排序的结果。

```

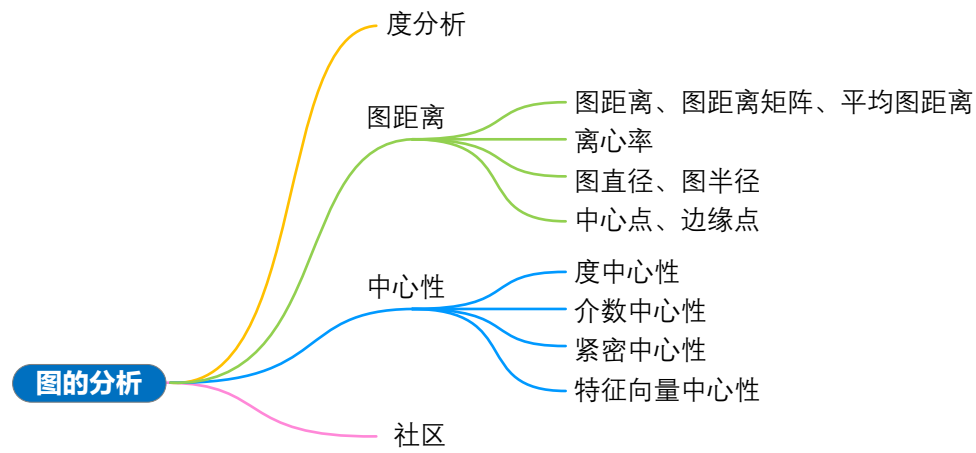
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

17.1 度分析

简单来说，**度分析** (degree analysis) 就是使用图的节点度数帮助我们分析图的连通性。

首先对空手道俱乐部图 (图 1 (a)) 进行度分析。图 1 (b) 所示为根据节点度数值大小用颜色映射渲染节点；暖色表示节点度数高，冷色代表节点度数低。显然，节点 0、33、32 在所有节点中度数相对较高。

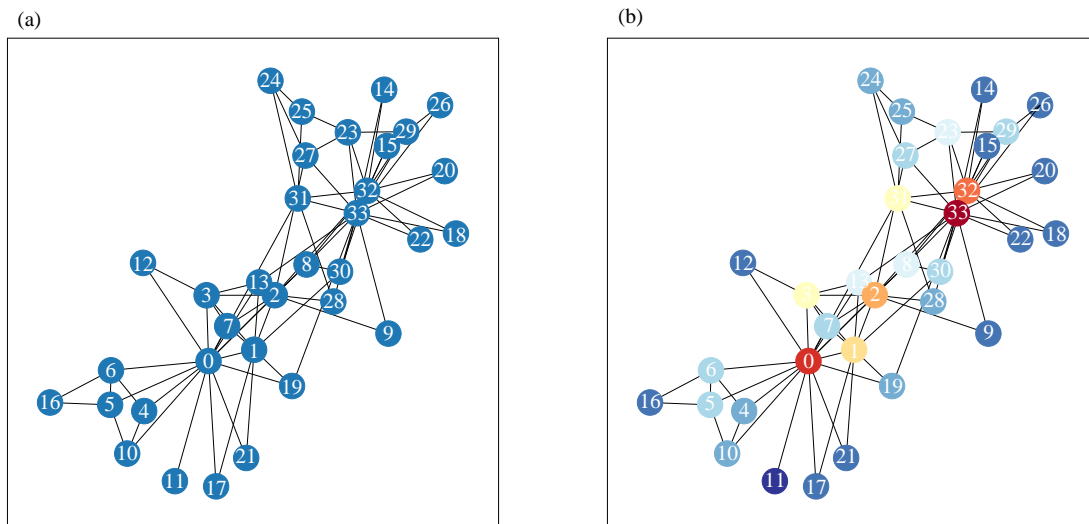


图 1. 空手道俱乐部会员关系图，根据节点度数渲染节点

图 2 用柱状图展示所有节点度数，显然所有节点中 34 的度数最高，0 紧随其后，32 的度数也不低。为了方便看到度数排序，我们还可以采用图 3 (a) 这种可视化方案；此外，图 3 (b) 还用柱状图展示节点度数分布情况。

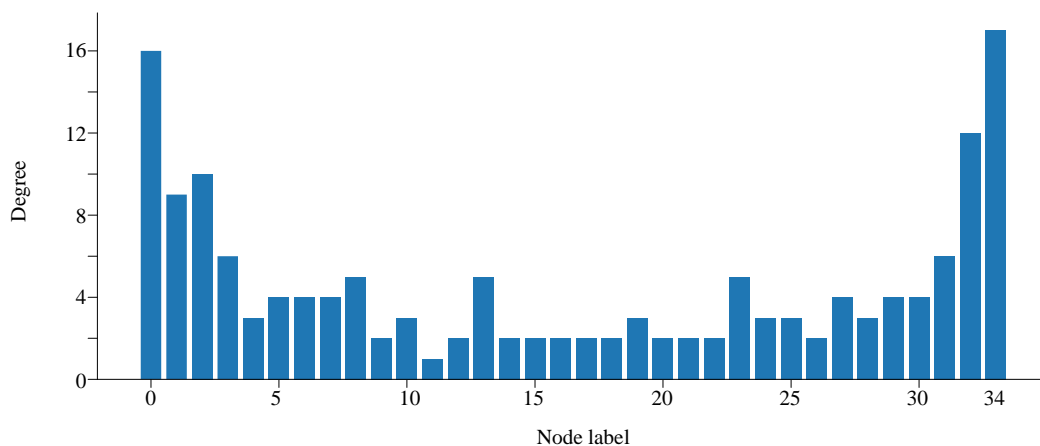


图 2. 空手道俱乐部会员关系图度分析，各个节点度数

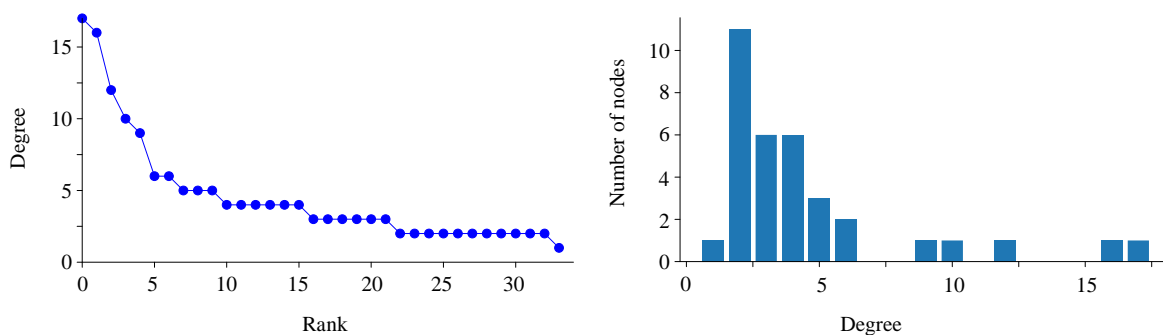


图 3. 空手道俱乐部会员关系图度分析，节点度数排序、节点度数柱状图

Bk6_Ch17_01.ipynb 完成空手道俱乐部会员关系图度分析，并绘制图 1 ~ 图 3。Bk6_Ch17_01.ipynb 和 Bk6_Ch17_03.ipynb 类似，本节只讲 Bk6_Ch17_03.ipynb。

Bk6_Ch17_03.ipynb 用来完成更复杂图的度分析。

图 4 (a) 是一幅有 100 个节点的图，图 4 (b) 绘制其中最大连通分量。

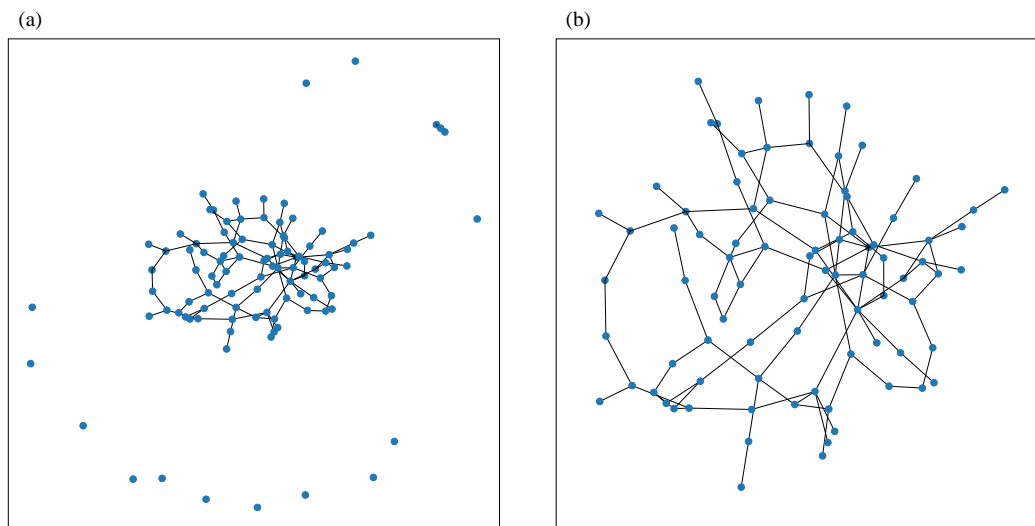


图 4. 图和最大连通分量

代码 1 绘制图 4 (b)，下面聊聊其中关键语句。注意，为了节省篇幅代码 1 仅仅给出部分代码，完整代码请大家参考配套文件 Bk6_Ch17_03.ipynb。

- a 用 `networkx.gnp_random_graph()` 创建图，有 100 个节点。
- b 先用 `networkx.connected_components(G)` 取出图 G 的连通分量，然后用 `sorted()` 根据连通分量节点数多少排序，再取出节点最大的连通分量。最后，用 `subgraph()` 方法创建子图。
- c 取出子图的节点坐标，保证图 4 (a) 和图 4 (b) 同一子图坐标一致。
- d 用 `networkx.draw_networkx()` 绘制子图。

```

import networkx as nx
import numpy as np
import matplotlib.pyplot as plt

a G = nx.gnp_random_graph(100, 0.02, seed=8)
# 创建随机图

# 连通分量 (节点数最多)
b Gcc = G.subgraph(sorted(nx.connected_components(G),
                           key=len, reverse=True)[0])

c pos_Gcc = {k: pos[k] for k in list(Gcc.nodes())}
# 取出子图节点坐标

# 可视化
plt.figure(figsize = (6,6))
d nx.draw_networkx(Gcc, pos_Gcc,
                    with_labels = False,
                    node_size=20)
plt.savefig('最大连通分量.svg')

```

代码 1. 连通分量 | Bk6_Ch17_03.ipynb

图 5 展示的是图 4 (a) 节点度数排序和柱状图。

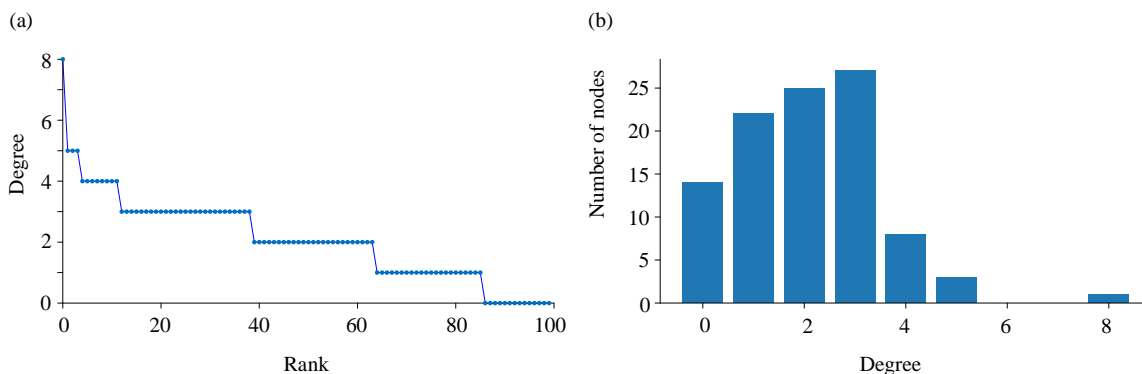


图 5. 度分析，节点度数排序、节点度数柱状图

接着前文代码，代码 2 对图进行度分析，下面聊聊其中关键语句。

- a 将图 G 各个节点度数取出，并从大到小排序。
- b 将图 G 各个节点度数取出，结果转化为字典。
- c 绘制线图展示从大到小排序的节点度数，如图 5 (a) 所示。
- d 绘制柱状图可视化节点度数分布，如图 5 (b) 所示。

```

# 度分析
a degree_sequence = sorted((d for n, d in G.degree()),
                           reverse=True)

# 度数大小排序

b dict_degree = dict(G.degree())
# 将结果转为字典

# 可视化度分析
c fig, ax = plt.subplots(figsize = (6,3))
ax.plot(degree_sequence, "b-", marker="o")
ax.set_ylabel("Degree")
ax.set_xlabel("Rank")
ax.set_xlim(0,100)
ax.set_ylim(0,8)
plt.savefig('度数等级图.svg')

d fig, ax = plt.subplots(figsize = (6,3))
ax.bar(*np.unique(degree_sequence, return_counts=True))
ax.set_xlabel("Degree")
ax.set_ylabel("Number of Nodes")
plt.savefig('度数柱状图.svg')

```

代码 2. 度分析 |  Bk6_Ch17_03.ipynb

图 6 所示为根据度数渲染节点。

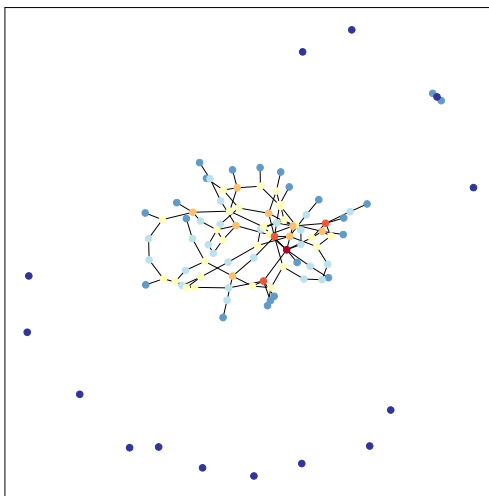


图 6. 根据度数渲染节点

接着前文代码，代码 3 绘制图 6，下面聊聊其中关键语句。

```

# 自定义函数，过滤dict
a def filter_value(dict_, unique):

    newDict = {}
    for (key, value) in dict_.items():
        if value == unique:
            newDict[key] = value

    return newDict

# 根据度数大小渲染节点
b unique_deg = set(degree_sequence)
# 取出节点度数独特值

c colors = plt.cm.RdYlBu_r(np.linspace(0, 1, len(unique_deg)))
# 独特值的颜色映射

d plt.figure(figsize = (6,6))
nx.draw_networkx_edges(G, pos)
# 绘制图的边

# 分别绘制不同度数节点
e for deg_i, color_i in zip(unique_deg, colors):

    dict_i = filter_value(dict_degree, deg_i)
    nx.draw_networkx_nodes(G, pos,
                           nodelist = list(dict_i.keys()),
                           node_size=20,
                           node_color = color_i)

plt.savefig('根据度数大小渲染节点.svg')

```

代码 3. 根据度数渲染节点 | BK6_Ch17_03.ipynb

a 自定义函数，用来从字典中提取 value 满足特定要求的部分，结果还是一个字典。请大家尝试用 filter() 函数重写这个函数。

b 用集合运算提取节点度数的独特值。

c 节点度数独特值的颜色映射；度数高用暖色调，度数低用冷色调。

d 用 networkx.draw_networkx_edges() 绘制图的边。

e 创建 for 循环根据节点度数大小分批绘制节点。为了避开这个 for 循环，大家可以尝试生成一个节点颜色映射 list。这样只需要调用 networkx.draw_networkx_nodes() 一次。

17.2 距离度量

图距离

在图论中，**图距离** (graph distance) 是指两个节点之间的最短路径的长度。在无权图中，图距离表示两个节点之间的最短路径的边数。如图 7 (a) 所示，节点 u 、 v 之间的距离为 1。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

在有权图中，图距离表示两个节点之间的最短路径的权重之和。

图距离用于衡量图中节点之间的距离或相似性，是图的基本性质之一。

图距离的计算对于许多图论和网络分析的任务非常重要。例如，社交网络中的两个用户之间的图距离可能表示它们之间的关系强度，而在交通网络中，两个地点之间的图距离可能表示最短行车路径的长度。图距离的计算也在路由算法、网络可达性分析等领域发挥着关键作用。

如图 7 (b)，所示节点 u 、 v 的距离为 2，两者之间可以有多个最短路径。如果没有连接两个节点的路径，则距离通常定义为无穷大。

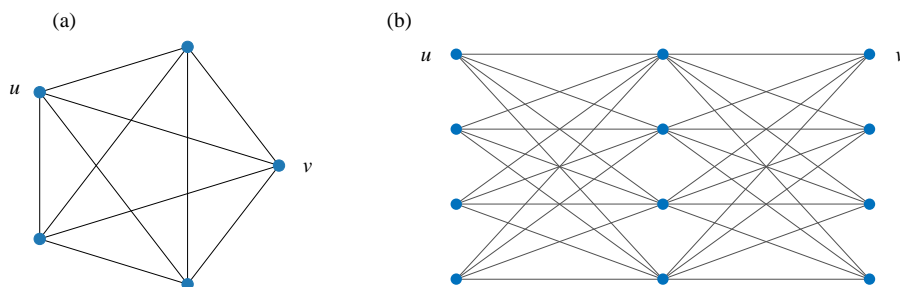


图 7. 图距离

图 8 所示为空手道俱乐部会员关系图 (不考虑权重) 中，节点 15、16 之间的图距离 $d_{15,16} = 5$ 。大家是否立刻想到任意两个节点之间都有图距离。

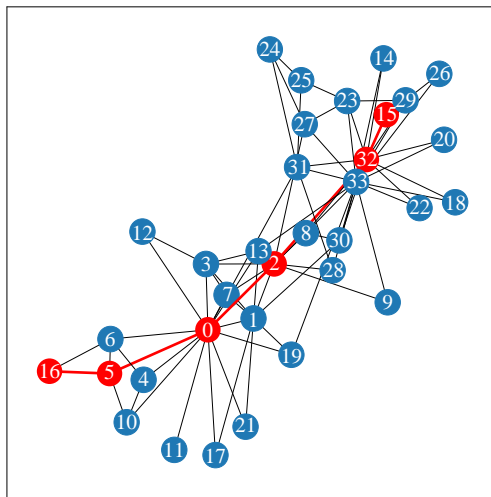


图 8. 空手道俱乐部会员关系图，节点 15、16 的图距离

代码 4 绘制图 8，下面聊聊其中关键语句。

- a** 用 `networkx.shortest_path()` 找到图中两节点之间最短路径。
- b** 用 `networkx.utils.pairwise()` 用于在路径节点序列中获取相邻元素的配对获得路径的边序列；参数 `cyclic=False` 为默认，当参数 `cyclic=True` 生成首尾闭合路径边序列。
- c** 用 `networkx.draw_networkx_nodes()` 绘制最短路径节点，节点颜色为红色。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

d 用 `networkx.draw_networkx_edges()` 绘制最短路径边，边颜色为红色。

```
import networkx as nx
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

G = nx.karate_club_graph()
# 空手道俱乐部图
pos = nx.spring_layout(G, seed=2)

a path_nodes = nx.shortest_path(G, 15, 16)
# 节点15、16之间最短路径

b path_edges = list(nx.utils.pairwise(path_nodes))
# 路径节点序列转化为边序列（不封闭）

plt.figure(figsize = (6,6))

nx.draw_networkx(G, pos)
c nx.draw_networkx_nodes(G, pos,
                        nodelist = path_nodes,
                        node_color = 'r')
d nx.draw_networkx_edges(G, pos,
                        edgelist = path_edges,
                        edge_color = 'r')

plt.savefig('空手道俱乐部图, 15、16最短路径.svg')
```

代码 4. 图距离 | Bk6_Ch17_02.ipynb

图距离矩阵

图 9 所示为图中所有成对距离构成的柱状图，纵轴为频数。而成对距离矩阵就是呈现这些图距离的最好方法！

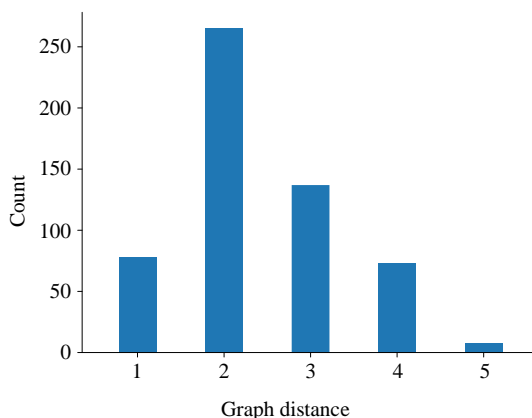


图 9. 空手道俱乐部会员关系图，成对图距离柱状图

图距离矩阵 (graph distance matrix, all-pairs shortest path matrix) 就是一张图每一对节点之间的图距离构造的矩阵。该矩阵提供了图中节点之间的所有可能路径的距离信息，对于图的全局结构和节点间的关系有着重要的信息。

图距离矩阵中的元素 d_{ij} 表示从节点 i 到节点 j 的最短路径长度。如果节点 i 和 j 之间没有直接的路径，那么 d_{ij} 可以被设定为无穷大。

图 10 空手道俱乐部会员关系图对应的图距离矩阵。图中对角线元素均为 0，代表节点到自身图距离。

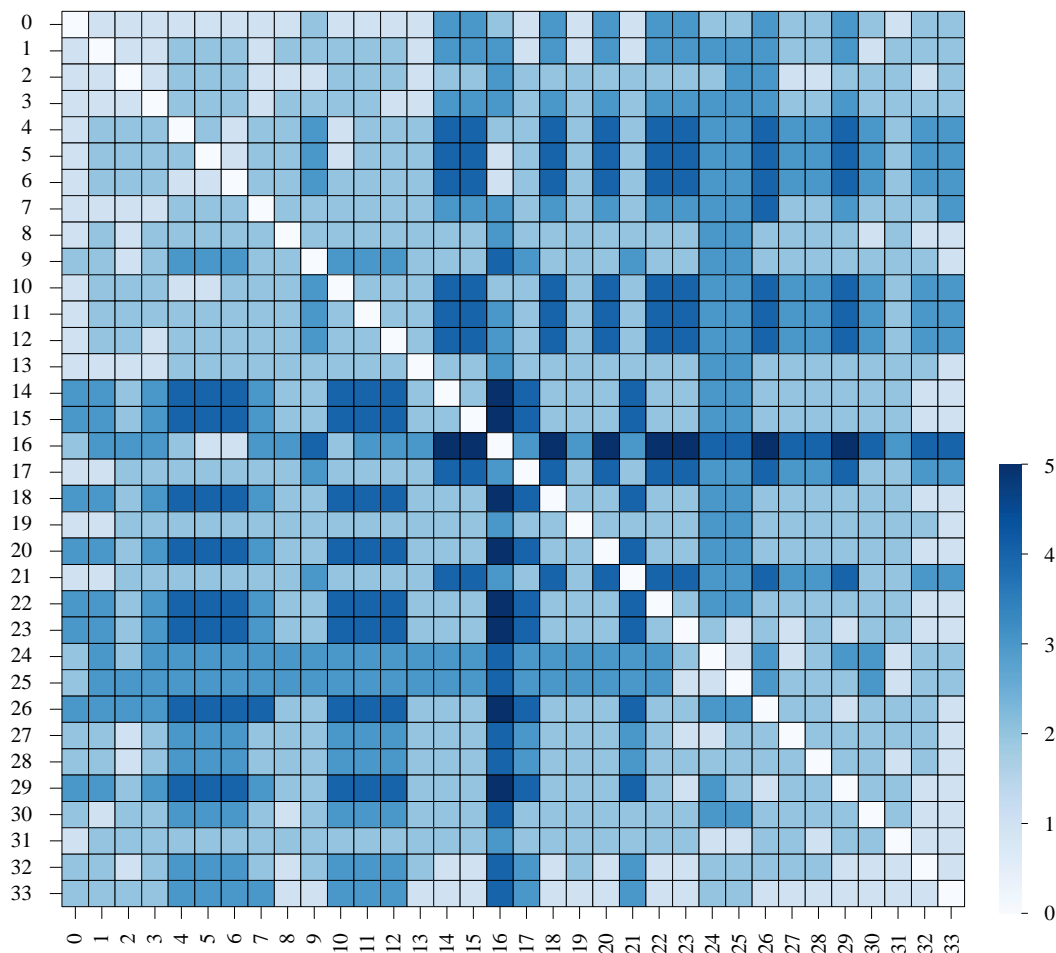


图 10. 空手道俱乐部会员关系图，图距离矩阵

图 10 也告诉我们任意节点和其他节点都存在图距离；对于任意节点，我们可以计算这些距离的平均值，叫做节点**平均图距离** (average graph distance)。

从图 10 来看，就是每行或每列所有元素 (对角线以外) 取均值。图 11 所示为空手道俱乐部会员关系图各个节点平均图距离。图 11 中水平红色划线为整幅图距离平均值，即所有节点平均图距离的平均值。

图 13 所示为根据节点平均图距离渲染节点。暖色系节点代表节点平均图距离较大，也就是说这些节点更“边缘”；相反冷色系代表节点平均距离更小，即这群节点更“中心”。本章后文还会介绍其他度量节点更中心或更边缘的度量。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

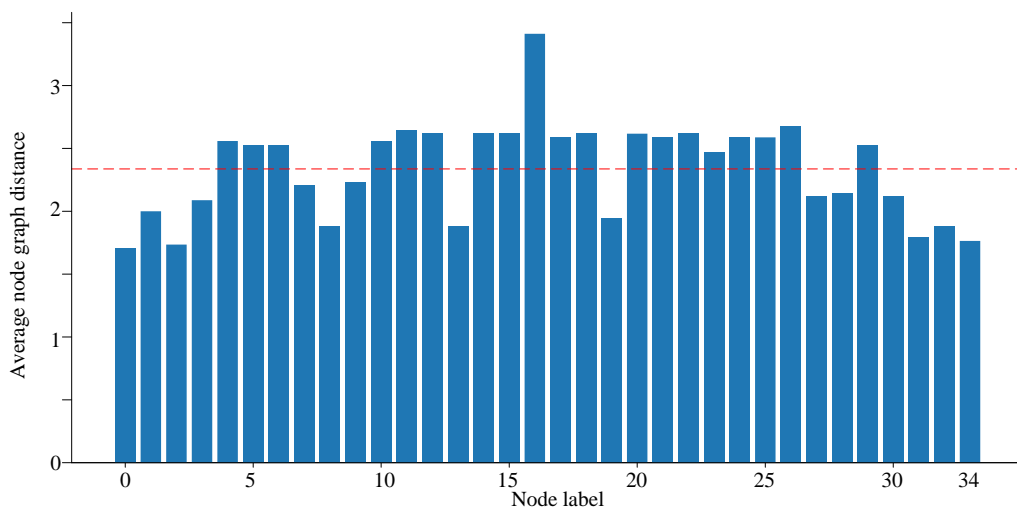


图 11. 空手道俱乐部会员关系图，各个节点平均图距离

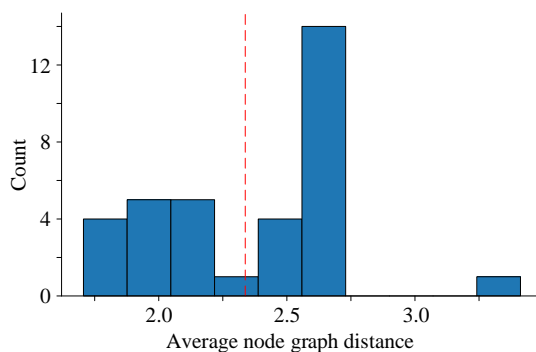


图 12. 空手道俱乐部会员关系图，节点平均图距离直方图

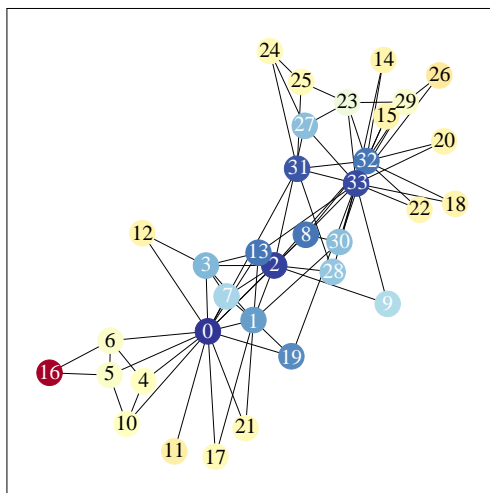


图 13. 空手道俱乐部会员关系图，用节点平均图距离渲染节点

节点平均图距离是一个有用的度量，它反映了一个节点与图中所有其他节点之间的平均距离。这个度量可以帮助我们理解一个节点在整个网络中的位置以及它与其他节点的相对接近度。在图论和网络分析中，节点平均图距离提供了对网络连通性和结构特性的洞察。

平均图距离较小的节点通常在网络中处于更中心的位置，表明它们可能在信息传播或网络流动中起着关键作用。整个网络的平均图距离，可以反映网络的整体效率和紧密度。

数值较小意味着网络中的节点相互之间更容易到达，表明网络具有较高的传播效率。通过比较不同节点或不同网络的平均图距离，可以揭示网络结构的特性和潜在的结构变化。

代码 5 绘制图 9 和图 10，下面聊聊其中关键语句。

a `networkx.shortest_path_length(G)` 用来图 G 中所有成对节点之间最短路径长度（图距离）。如果没有指定 `source` 和 `target`，函数返回一个字典，其中键是源节点，值是另一个字典，该字典的键是目标节点，值是从源节点到目标节点的最短路径长度。下文代码会把这个嵌套字典转化为二维数组。

b 用嵌套 `for` 循环将嵌套字典转换为图距离矩阵，下面聊聊其中细节。第一个 `for` 循环遍历 `list_nodes` 列表中的所有节点，相当于始点。第二层 `for` 循环相当于遍历所有终点。

c 用 `try ... except ...` 处理可能不存在路径的情况。

d 用 `seaborn.heatmap()` 绘制热图可视化图距离矩阵。

e 使用 `numpy.tril()` 获取下三角矩阵，并排除对角线元素。

f 获取下三角矩阵（不含对角线）的索引。

g 使用索引从原矩阵中取出对应的元素，这是因为图距离为对称矩阵，节点成对距离（非对角线元素）重复。

h 使用 `numpy.unique()` 函数获取图距离独特值及其出现次数。

```

# 成对最短距离值（图距离）
a distances_all = dict(nx.shortest_path_length(G))

# 创建图距离矩阵
list_nodes = list(G.nodes())
Shortest_D_matrix = np.full((len(G.nodes()),
                             len(G.nodes())), np.nan)

# 遍历所有节点对，计算最短距离
b for i, i_node in enumerate(list_nodes):
    for j, j_node in enumerate(list_nodes):
        c try:
            d_ij = distances_all[i_node][j_node]
            Shortest_D_matrix[i][j] = d_ij
        except KeyError:
            print(i_node + ' to ' + j_node + ': no path')

Shortest_D_matrix.max()
# 图距离最大值

# 用热图可视化图距离矩阵
d sns.heatmap(Shortest_D_matrix, cmap = 'Blues',
              annot = False,
              xticklabels = list(G.nodes()),
              yticklabels = list(G.nodes()),
              linecolor = 'k', square = True,
              cbar = True,
              linewidths = 0.2)
plt.savefig('图距离矩阵, 无权图.svg')

# 取出图距离矩阵中所有成对图距离（不含对角线下三角元素）

# 使用numpy.tril获取下三角矩阵，并排除对角线元素
e lower_tri_wo_diag = np.tril(Shortest_D_matrix, k=-1)

# 获取下三角矩阵（不含对角线）的索引
f rows, cols = np.tril_indices(Shortest_D_matrix.shape[0], k=-1)

# 使用索引从原矩阵中取出对应的元素
g list_shortest_distances = Shortest_D_matrix[rows, cols]

# 使用numpy.unique函数获取唯一值及其出现次数
h unique_values, counts = np.unique(list_shortest_distances,
                                     return_counts=True)

# 绘制柱状图
plt.bar(unique_values, counts)
plt.xlabel('Graph distance')
plt.ylabel('Count')
plt.savefig('图距离柱状图.svg')

```

代码 5. 图距离矩阵 | Bk6_Ch17_02.ipynb

离心率

图中任意节点 v 的**离心率** (eccentricity) 就是图中离 v 图距离的最大值。

图 14 空手道俱乐部会员关系图所有节点的离心率具体值；图 15 所示为该图离心率分布。

图 16 用根据离心率大小渲染节点，图中红色节点的离心率为 3，黄色节点的离心率为 4，蓝色对应离心率是 5。容易发现，节点越居于“中心”，对应的离心率越小；通俗地说，这个节点更“合群”，和其他节点距离都近，要么是朋友关系，要么通过朋友的朋友就可以相互认识。而节点越位于“边缘”，对应的离心率越大；也就说在这个社会关系里，离心率越高的节点在网络中越“不合群”。

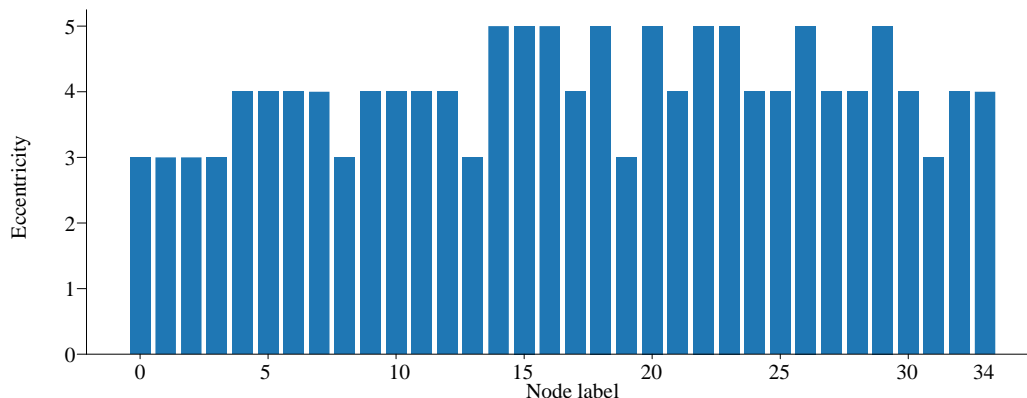


图 14. 空手道俱乐部会员关系图，各个节点离心率

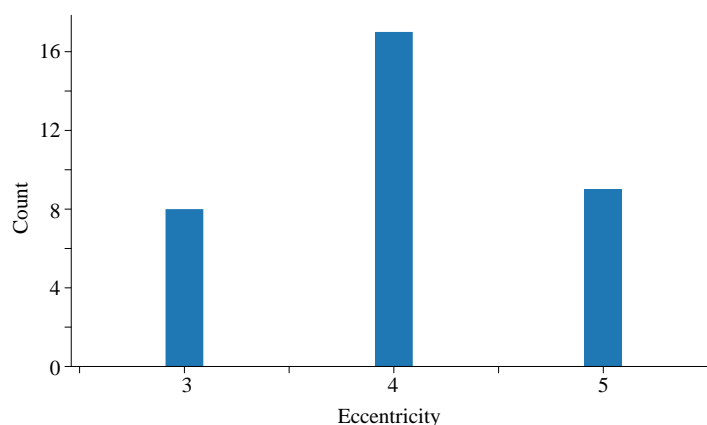
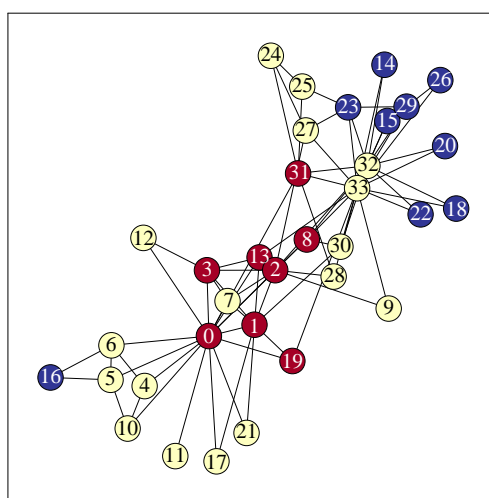


图 15. 空手道俱乐部会员关系图，离心率柱状图



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 16. 空手道俱乐部会员关系图，根据节点离心率渲染节点

代码 6 绘制图 14、图 15、图 16，下面聊聊其中关键语句。

- a 用 `networkx.eccentricity(G)` 计算每个节点的离心率。
- b 取出节点离心率独特值。
- c 独特值的颜色映射。
- d 绘制不同离心率节点，用不同颜色渲染节点。
- e 获取离心率独特值以及出现的次数。

```
a eccentricity = nx.eccentricity(G)
# 计算每个节点离心率
eccentricity_list = list(eccentricity.values())

# 根据离心率大小渲染节点

b unique_ecc = set(eccentricity_list)
# 取出节点离心率独特值

c colors = plt.cm.RdYlBu(np.linspace(0, 1, len(unique_ecc)))
# 独特值的颜色映射

plt.figure(figsize = (6,6))
nx.draw_networkx_edges(G, pos)
# 绘制图的边

# 分别绘制不同离心率节点
d for deg_i, color_i in zip(unique_ecc, colors):
    dict_i = filter_value(eccentricity, deg_i)
    nx.draw_networkx_nodes(G, pos,
                           nodelist = list(dict_i.keys()),
                           node_color = color_i)
plt.savefig('根据离心率大小渲染节点.svg')

# 每个节点的具体离心率
plt.bar(G.nodes(), eccentricity_list)
plt.xlabel('Node label')
plt.ylabel('Eccentricity')
plt.savefig('节点离心率.svg')

# 使用numpy.unique函数获取离心率独特值及其出现次数
e unique_values, counts = np.unique(eccentricity_list,
                                    return_counts=True)

# 绘制柱状图
plt.bar(unique_values, counts)
plt.xlabel('Eccentricity')
plt.ylabel('Frequency')
plt.savefig('图离心率柱状图.svg')
```

代码 6. 离心率 | Bk6_Ch17_02.ipynb

图直径

图距离矩阵所有元素最大值叫做**图直径** (graph diameter)，也就是 longest shortest path。非连通图的直径无穷大。显然，所有节点离心率的最大值就是图直径。

根据这个定义，图 15 便告诉我们空手道俱乐部会员关系图的图直径为 5。

直观地说，图直径是一个衡量图的大小和稀疏程度的指标。图直径提供了对图整体大小和节点之间最远距离的感知。图直径的大小反映了图的大致尺寸。一幅图如果直径较大，这说明图中存在一些较为疏远的节点。在一些网络分析和图算法中，图直径被用作衡量图的全局性质的一个指标。

图半径

所有节点离心率的最小值叫**图半径** (graph radius)。图半径可以用来描述图的“紧凑性”。半径越小，表示网络中任意两点之间的最远距离越短，网络越紧凑。在网络设计和网络分析中，了解网络的图半径可以帮助设计更有效的网络结构，比如提高网络的传输效率，减少延迟等。在社交网络分析中，图半径可以用来衡量信息传播的最大延迟，或者找到网络中的关键人物。

根据这个定义，图 15 同时告诉我们空手道俱乐部会员关系图的图半径为 3。

中心点

图中节点 v 要是**中心点** (center) 的话， v 的离心率等于图半径；也就是 v 在所有节点中离心率最小。一幅图中中心点不止一个。如图 17 所示，红色节点都是这幅图的中心点。**图中心** (graph center) 就是图的中心点构成的集合。

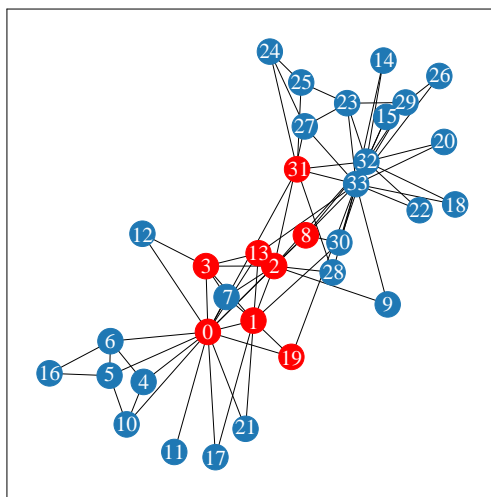


图 17. 空手道俱乐部会员关系图，中心点

代码 7 绘制图 17，其中 `networkx.center()` 计算图的中心点。


```

a list_centers = list(nx.center(G))
# 获取图的中心点

plt.figure(figsize = (6,6))

nx.draw_networkx(G, pos)
nx.draw_networkx_nodes(G, pos,
                       nodelist = list_centers,
                       node_color = 'r')
plt.savefig('空手道俱乐部图, 中心点.svg')

```

代码 7. 中心点 | Bk6_Ch17_02.ipynb

边缘点

和中心点相对，一幅图的**边缘点** (peripheral point) 是离心率为图直径的点。也就是说，这些节点的离心率最大。同样，一幅图中中心点不止一个。如图 18 所示，红色节点都是这幅图的边缘点。一幅图的边缘点构成的子图叫做**图边缘** (graph periphery)。

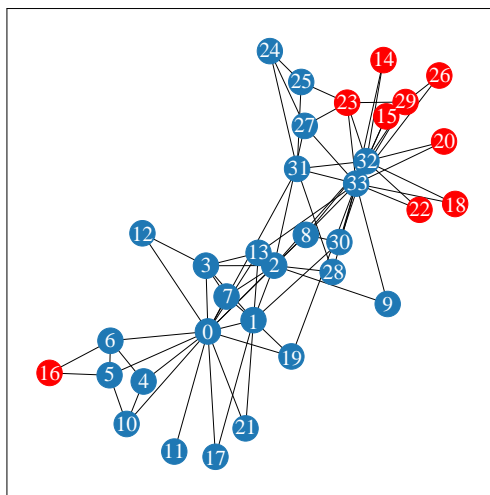


图 18. 空手道俱乐部会员关系图，边缘点

代码 8 绘制图 18，其中 `networkx.periphery()` 计算图的边缘点。

```

a list_periphery = list(nx.periphery(G))
# 获取图的边缘点

plt.figure(figsize = (6,6))

nx.draw_networkx(G, pos)
nx.draw_networkx_nodes(G, pos,
                       nodelist = list_periphery,
                       node_color = 'r')
plt.savefig('空手道俱乐部图, 边缘点.svg')

```

代码 8. 边缘点 |  Bk6_Ch17_02.ipynb

17.3 中心性

中心性 (centrality) 用来描述节点在图有多“中心”，本节介绍三个常用的中心性度量。

度中心性

度中心性 (degree centrality) 用节点的度数描述其“中心性”。简单来说，一个节点度数越大就意味着这个节点的度中心性越高，该节点在网络中就越重要。

我们可以通过本章前文介绍的度分析来计算整个图的最大度、最小度、平均度来衡量整个图的中心性。

对于无向图，节点 a 的度中心可以通过下式计算

$$\frac{\deg_G(a)}{n-1} \quad (1)$$

$\deg_G(a)$ 表示节点 a 的度数， n 代表图的节点个数。上式的好处是通过归一化，方便不同图之间比较。当然，对于多重边的图，度中心可能大于 1。

以图 19 为例，节点 a 、 b 、 c 、 d 、 e 的度数分别是 1、2、2、2、1，无向图的节点数 $n=5$ 。各个节点度数除以 4 ($n-1=4$)，得到五个节点度中心性度量值分别为 0.25、0.5、0.5、0.5、0.25。



图 19. 计算无向图度中心性，链图

NetworkX 中，我们可以用 `networkx.degree_centrality()` 计算度中心性。

对于有向图 G_D ，我们可以分别分析节点 a 的入度中心性、出度中心性：

$$\frac{\deg_{G_D}^+(a)}{n-1} \quad (2)$$

$$\frac{\deg_{G_D}^-(a)}{n-1}$$

NetworkX 中，我们可以用 `networkx.in_degree_centrality()`、`networkx.out_degree_centrality()` 计算入度中心性、出度中心性。

图 20 所示为空手道俱乐部会员关系图，以及对应的度中心性度量值。图 20 (a) 用直方图可视化度中心性度量值的分布情况。图 20 (b) 则根据度中心度量值大小渲染节点，采用的颜色映射为 `viridis`。

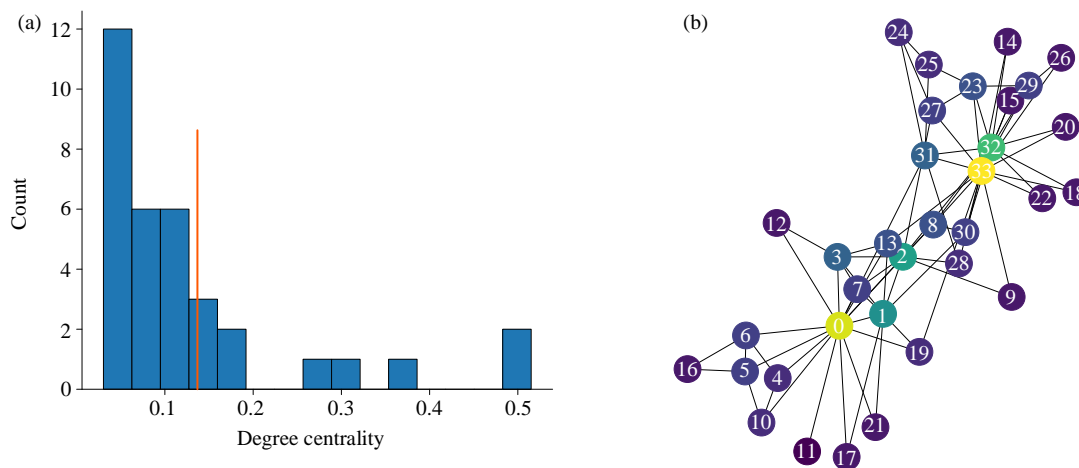


图 20. 度中心性，空手道俱乐部会员关系图

Bk6_Ch17_04.ipynb 完成本节空手道俱乐部会员关系图的中心性计算，下面先聊聊代码 9 这段代码中的关键语句。

- a** 自定义可视化函数，子图左右布置；左侧子图为度中心性度量的直方图，右侧子图可视化无向图，节点颜色根据中心性度量值大小渲染。
- b** 用 `matplotlib.pyplot.hist()` 绘制直方图。
- c** 先将中心性度量值转换成 NumPy 数组，然后再计算平均值。
- d** 用 `matplotlib.pyplot.axvline()` 绘制均值竖线。
- e** 用 `networkx.draw_networkx()`，显示节点标签，节点颜色根据中心性度量值渲染。
- f** 用 `networkx.karate_club_graph()` 加载空手道俱乐部会员关系图数据。
- g** 用 `networkx.degree_centrality()` 计算度中心性。

```

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import networkx as nx

# 自定义可视化函数
def visualize(x_cent, xlabel):

    fig, axes = plt.subplots(1,2,figsize = (12,6))

    # 中心性度量值直方图
    axes[0].hist(x_cent.values(),bins = 15, ec = 'k')
    # 中心性度量值均值
    mean_cent = np.array(list(x_cent.values())).mean()
    axes[0].axvline(x = mean_cent, c = 'r')
    axes[0].set_xlabel(xlabel)
    axes[0].set_ylabel('Count')

    degree_colors = [x_cent[i] for i in range(0,34)]

    # 可视化图，用中心性度量值渲染节点颜色
    nx.draw_networkx(G, pos,
                    ax = axes[1],
                    with_labels = True,
                    node_color=degree_colors)

    plt.savefig(xlabel + '.svg')

f G = nx.karate_club_graph()
# 加载空手道俱乐部数据

pos = nx.spring_layout(G, seed=2)

g degree_cent = nx.degree centrality(G)
# 计算度中心性

# 可视化
visualize(degree_cent, 'Degree centrality')

```

代码 9. 度中心性度量 |  Bk6_Ch17_04.ipynb

介数中心性

介数中心性 (betweenness centrality) 量化节点在图中承担“桥梁”程度。

具体来说，某个节点 v 介数中心性计算 v 出现在其他任意两个节点对 (s,t) 之间的最短路径的次数，本书采用 NetworkX 的定义，具体如下

$$\sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)} \quad (3)$$

V 是无向图节点集合。 (s,t) 是无向图中任意一对节点。

上式分母 $\sigma(s,t)$ 代表无向图中节点对 (s,t) 最短路径的总数。特别地，如果 $s = t$ ， $\sigma(s,t) = 1$ 。

上式分子 $\sigma(s,t|v)$ 代表无向图中所有最短路径中 (排除首尾 s 和 t) 含有 v 的数量。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

这个中心性度量设计成 (3) 这种分数的形式是因为节点对 (s, t) 最短路径可能不止一个。

也就是说，如果 v 在任意两个节点间充当“桥梁”的次数越多，那么 v 的介数中心性就越大。

对于图 19 所示链图，图 21 给出在 $s \neq t$ 条件下，所有最短路径。绿色点代表节点充当“桥梁”的情况。

我们可以发现节点 a 、 b 、 c 、 d 、 e 充当“桥梁”的情况数量分别为 0、3、4、3、0，即介数中心性度量值大小。

用 `networkx.betweenness centrality(G, normalized = False)` 计算图 G 的介数中心性会得出上述结果，参数 `normalized = False` 表示不归一化。

默认情况，`normalized = True`，`networkx.betweenness centrality(G)` 计算结果就是**归一化介数中心性** (normalized betweenness centrality)。

NetworkX 中对于无向图，归一化的乘数为 $\frac{2}{(n-1)(n-2)}$ ；其中 n 是图的节点数。这样对于图 19 链图

归一化乘数为 $\frac{2}{4 \times 3} = \frac{1}{6}$ ，节点 a 、 b 、 c 、 d 、 e 的归一化介数中心性度量值分别为 0、0.5 (3/6)、2/3 (4/6)、0.5 (3/6)、0。

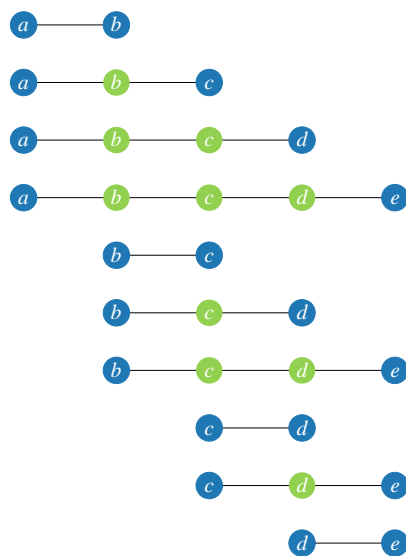


图 21. 链图所有最短路径， $s \neq t$

图 22 (a) 用直方图可视化归一化介数中心性度量值的分布情况。图 22 (b) 则根据归一化介数中心性度量值大小渲染节点。

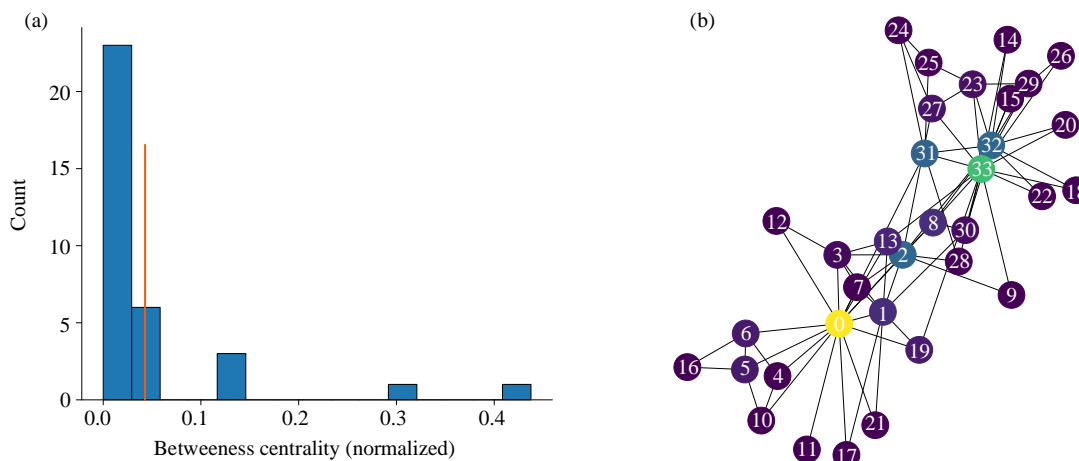


图 22. 归一化介数中心性，空手道俱乐部会员关系图

代码 10 计算介数中心性，下面简单聊聊如下两句。

- a 用 `networkx.betweenness centrality(G, normalized = False)` 计算介数中心性度量值。
- b 用 `networkx.betweenness centrality(G)` 计算归一化介数中心性度量值，默认参数 `normalized = True`。请大家自行比较两句结果。

```

a nx.betweenness centrality(G, normalized = False)
  # 介数中心性

b betweenness_cent = nx.betweenness centrality(G)
  # 计算归一化介数中心性

  # 可视化归一化介数中心性
visualize(betweenness_cent, 'Betweenness centrality')

```

代码 10. 介数中心性度量 | Bk6_Ch17_04.ipynb

图 23 展示基因之间的介数中心性，它使用了 WormNet v.3-GS 数据来测量基因之间的正向功能关联。WormNet 是一个用于研究秀丽隐杆线虫 (*Caenorhabditis elegans*) 基因功能网络的资源。

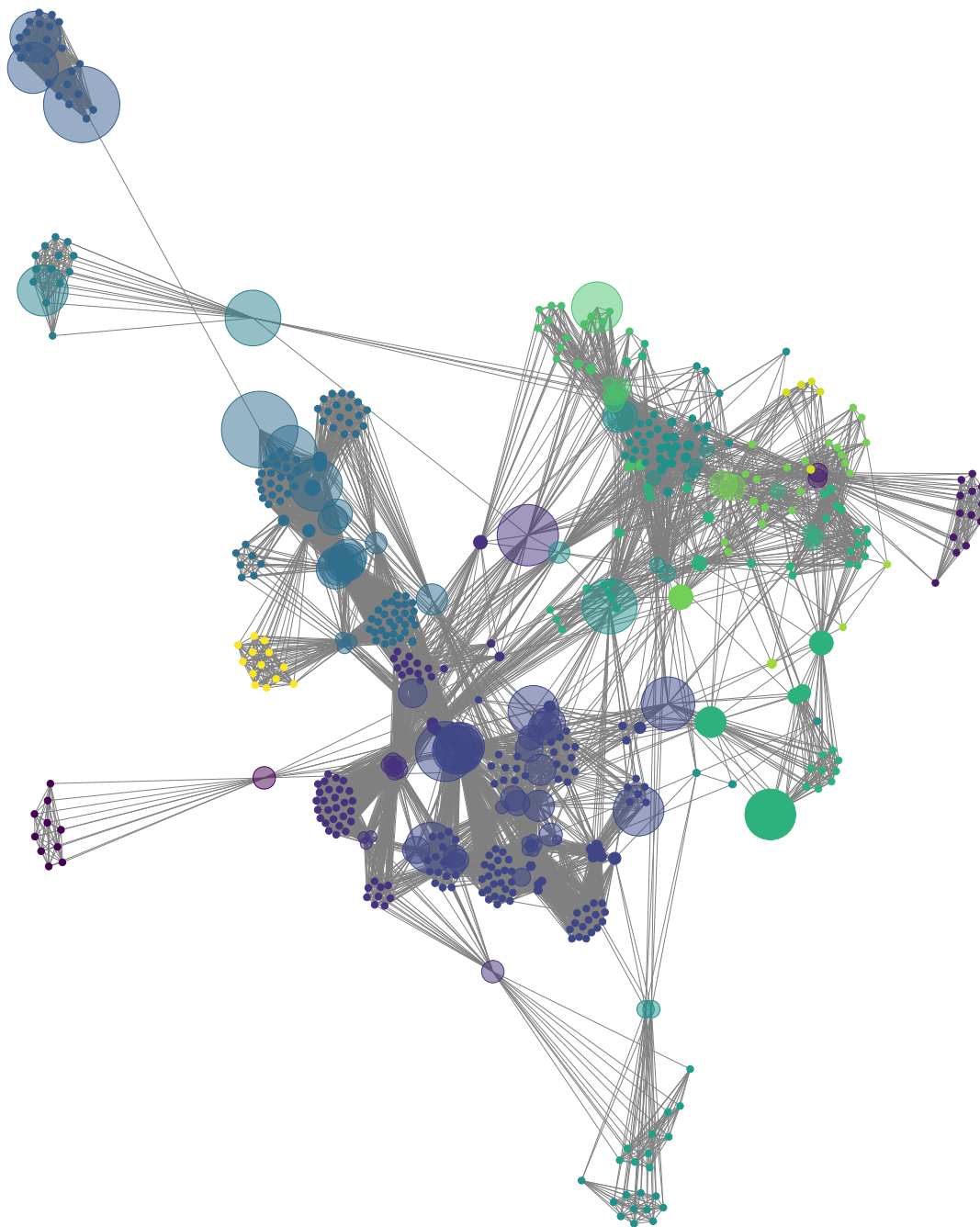


图 23. WormNet v.3-GS 数据来测量基因之间的正向功能关联

请大家自行学习：

https://networkx.org/documentation/stable/auto_examples/algorithms/plot_betweenness centrality.html

紧密中心性

对于节点 a ，其**紧密中心性** (clones centrality) 具体定义如下

$$\frac{k-1}{\sum_{v=1}^{k-1} d(a,v)} \quad (4)$$

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

其中， $d(a, v)$ 是节点 a 和 v 最短距离， $k-1$ 代表节点 a 可达节点的数量；也就是说 k 为包含节点自身可达节点数量。

将上式写成：

$$\frac{1}{\frac{\sum_{v=1}^{k-1} d(a, v)}{k-1}} \quad (5)$$

我们可以发现，分母是节点 a 和可达节点之间平均最短距离。

这说明，平均最短距离越大，越远离中心；取倒数的话，紧密中心性（平均最短距离倒数）越大，节点越靠近中心。

以链图中节点 a 为例，如图 24 所示，节点 a 可到达的节点（包含自身）为 $k=5$ ， $k-1=4$ 。

$\sum_{v=1}^{k-1} d(a, v)$ 的值为图 24 中所有距离之和，即 10。这样的话，节点 a 的紧密中心性度量值为 0.4 (= 4/10)。

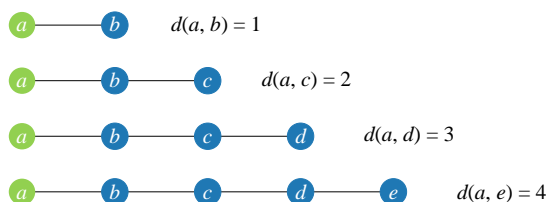


图 24. 计算节点 a 的紧密中心性，链图

再来计算节点 b 的紧密中心性度量值，具体如图 25 所示。节点 b 可到达的节点（包含自身）也是 $k=5$ ， $k-1=4$ 。 $\sum_{v=1}^{k-1} d(b, v)$ 的值为图 25 中所有距离之和，即 7。这样的话，节点 b 的紧密中心性度量值为 4/7。

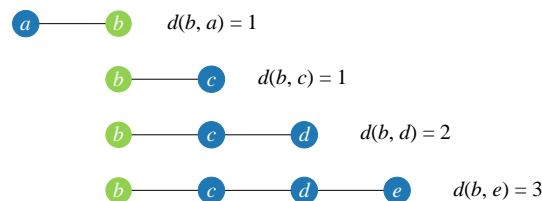


图 25. 计算节点 b 的紧密中心性，链图

请大家计算链图中剩余其他节点的紧密中心性度量值。

图 26 (a) 用直方图可视化紧密中心性度量值的分布情况。图 26 (b) 则根据紧密中心性度量值大小渲染节点。Bk6_Ch17_04.ipynb 利用 `networkx.closeness centrality()` 计算紧密中心性。

对于有向图，我们可以分别计算入度紧密中心性、出度紧密中心性。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

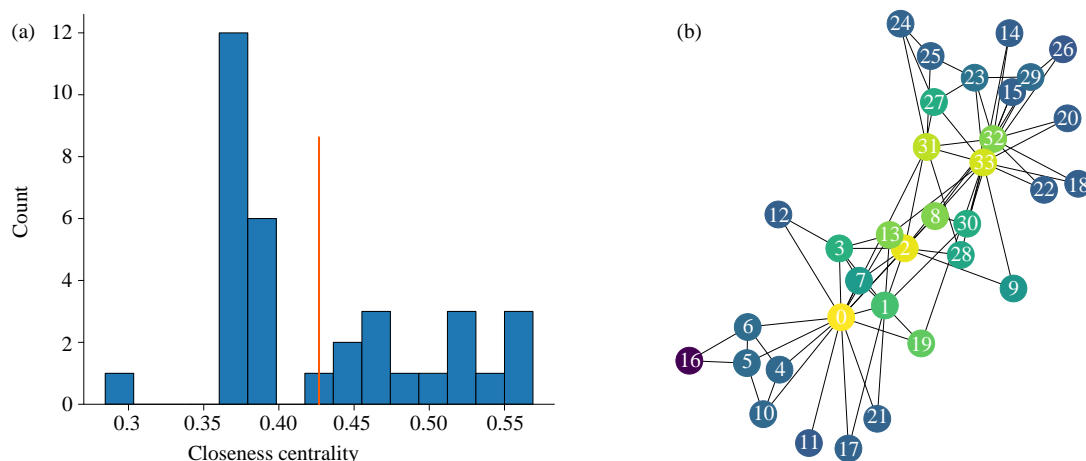


图 26. 紧密中心性，空手道俱乐部会员关系图

此外，Bk6_Ch17_04.ipynb 还计算如图 27 所示的**特征向量中心性** (eigenvector centrality)；这个中心度量要用到无向图的邻接矩阵、特征值分解等线性代数工具，这是本书后续要讲解的内容。

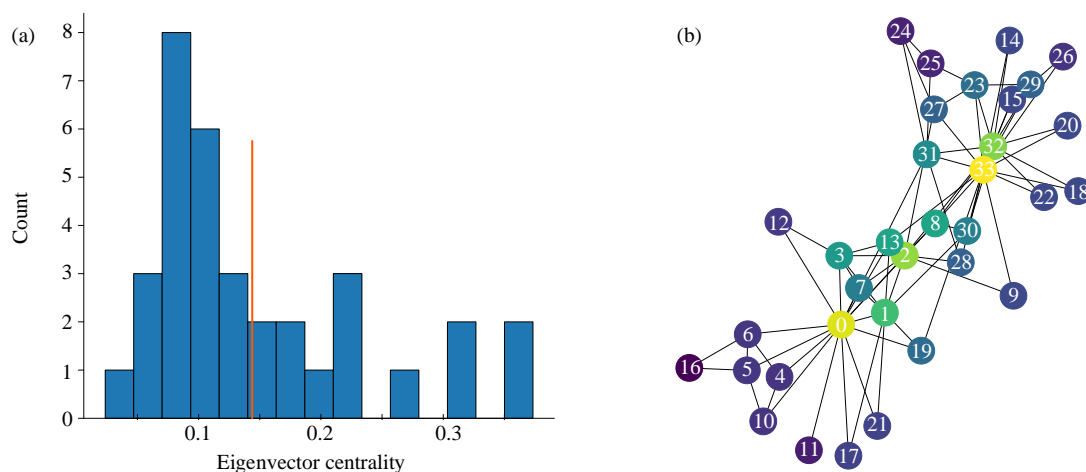


图 27. 特征向量中心性，空手道俱乐部会员关系图

17.4 图的社区

图中的一个**社群** (community) 可以这样理解，紧密连接的节点集合，这些节点间有较多的内部连接，而相对较少的外部连接。社区划分的应用有很多。比如，在蛋白质网络中，社区检测有助于发现相似生物学功能的蛋白质；在企业网络中，可以通过研究公司的内部关系将员工分组为社区；在在线平台社交网络中，具有共同兴趣或共同朋友的用户可能是同一个社区的成员。

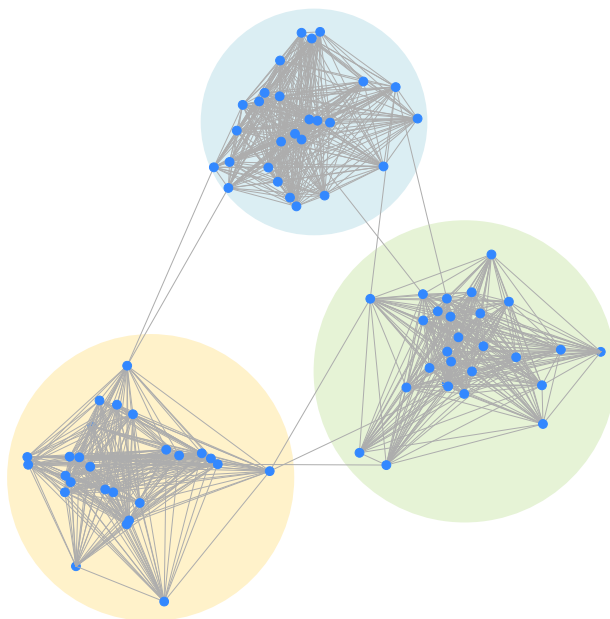


图 28. 一幅图中的三个社区，空手道俱乐部会员关系图

如图 29 所示，仅靠图的结构化关系，可以比较合理地将空手道俱乐部会员进行切分，即规划至各自的社群。

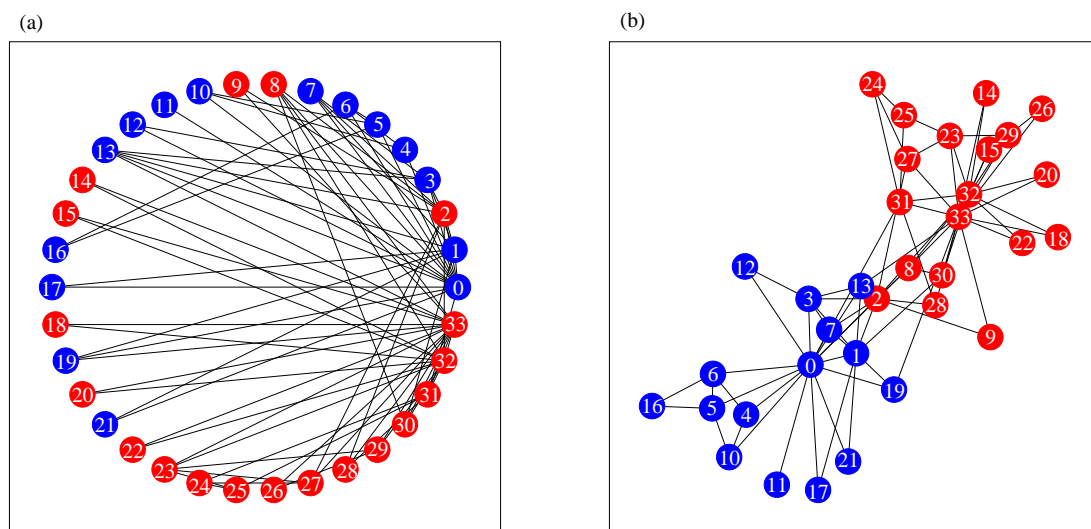
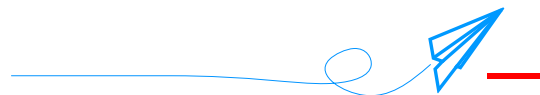


图 29. 两个社区，空手道俱乐部会员关系图

Bk6_Ch17_05.ipynb 完成空手道俱乐部会员关系图社区划分，并绘制图 29；代码比较简单，请大家自行学习。



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图的分析通过多个维度揭示网络结构的特性。

度分析关注节点度数来揭示最活跃或最重要的节点。

图距离是图中两节点间最短路径的长度。图距离矩阵记录了所有节点对间的图距离，是分析图结构的重要工具，便于快速查询任意两点间的最短距离。离心率是节点到图中所有其他节点最短路径长度的最大值，用以衡量节点的边缘性。图直径是所有节点对离心率的最大值；图半径是所有节点的离心率中的最小值。中心点是离心率等于图半径的所有节点，表明这些点在结构上最为核心。边缘点是离心率等于图直径的节点，位置最边缘，通常在网络的外围。这些概念在网络分析、优化路径、社交网络分析等领域中有广泛应用，帮助揭示网络的结构特性和关键节点。

中心性衡量节点在图中的重要性。度中心性通过连接数，介数中心性通过节点在最短路径上的出现频率，紧密中心性则考量节点到所有其他节点的平均最短路径长度，来识别关键节点。本书后续还要介绍特征向量中心性，请大家将这四种中心性度量放在一起比较。

社区发现旨在识别图中的紧密连接的节点群体，揭示网络中的模块结构或集团，帮助理解图的大规模结构特性。这些方法为理解和分析复杂网络提供了强有力的工具。

本章介绍的内容将用于本书后文的路径问题和社交网络分析。