Friend-based Ranking: Figures

This notebook is to produce figures for *Friend-based Ranking*, available at arXiv:1807.05093 (https://arxiv.org/pdf/1807.05093.pdf).

Setup

Packages

In [1]:

```
# Import packages
import numpy as np
import scipy as sp
import networkx as nx
import decimal
import math
import pandas as pd
import statsmodels.api as sm
import random

# To import Matlab matrices
import scipy.io

# Plotting
import matplotlib.pyplot as plt
from matplotlib.patches import Patch
import seaborn as sns
```

In [2]:

```
# Suppress the warnings from matplotlib about networkx
import warnings
warnings.filterwarnings("ignore")
# Pandas display settings
pd.set_option("display.max_rows",999)
pd.options.display.float_format = '{:,.4f}'.format
#Seaborn display settings
sns.set(style="ticks", palette="Greys", font_scale=1.4)
#Display plots inside notebook
%matplotlib inline
```

Generate the random seed from random.org (https://www.random.org/integers/?num=1&min=1&max=100000&col=1&base=10&format=html&rnd=new)

In [3]:

```
# Seed for random numbers
seed = 40588
random.seed(seed)
```

Colors

I use the ColorBrewer tool (http://colorbrewer2.org/) to choose color palettes.

```
In [4]: sns.color_palette("Greys", n_colors=2)

Out[4]: [(0.7777777777777778, 0.7777777777777778, 0.7777777777777778),
         (0.40784313725490196, 0.40784313725490196, 0.40784313725490196)]
```

```
In [5]: colors = {'Indonesia':'#66c2a5','India':'#fc8d62'}
        grey_light = '#bdbdbd'
        grey_dark = '#636363'
```

Read data

Save the panda dataframes to pickle files so that I don't need to extract the data again.

```
In [6]: df = pd.read_pickle('pd_df/netdata.pickle')
```

Tables

```
In [7]: df.columns

Out[7]: Index(['ave_dist', 'ave_clust', 'num_edges', 'ave_deg', 'diameter', 'density',
               'num_nodes', 'info_total', 'info_total_friend_only', 'info_expostIC',
               'info_SP', 'links_supported', 'comp_total', 'comp_supp', 'comp_trans',
               'comp_by_three', 'key', 'country', 'calculated_on'],
              dtype='object')
```

In [8]:
```
df.groupby(['country','calculated_on'])[['num_nodes','ave_deg','density',
                                          'ave_clust','ave_dist',
                                          'info_total','links_supported']].agg(['mean'
,'min','max','count']).transpose()
```

Out[8]:

	country	India		Indonesia	
	calculated_on	full	giant	full	giant
num_nodes	**mean**	198.7200	188.8667	52.9858	23.6019
	min	77.0000	75.0000	11.0000	2.0000
	max	356.0000	341.0000	263.0000	82.0000
	count	75.0000	75.0000	633.0000	633.0000
ave_deg	**mean**	8.8999	9.3437	3.8477	6.4979
	min	6.1091	6.8224	0.0769	1.0000
	max	13.4444	13.8286	15.8605	20.6667
	count	75.0000	75.0000	633.0000	633.0000
density	**mean**	0.0491	0.0541	0.1049	0.3588
	min	0.0225	0.0245	0.0012	0.0930
	max	0.1141	0.1208	0.8627	1.0000
	count	75.0000	75.0000	633.0000	633.0000
ave_clust	**mean**	0.2510	0.2638	0.4149	0.7320
	min	0.1516	0.1627	0.0000	0.0000
	max	0.4397	0.4535	0.9274	1.0000
	count	75.0000	75.0000	633.0000	633.0000
ave_dist	**mean**	nan	2.7463	1.5017	2.0432
	min	nan	2.2955	1.1373	1.0000
	max	nan	3.3163	1.9300	4.8097
	count	0.0000	75.0000	5.0000	633.0000
info_total	**mean**	0.3459	0.3805	0.2218	0.7142
	min	0.1716	0.1873	0.0018	0.2051
	max	0.5981	0.6306	1.0000	1.0000
	count	75.0000	75.0000	633.0000	633.0000
links_supported	**mean**	0.8182	0.8182	0.9350	0.9542
	min	0.6751	0.6751	0.0000	0.0000
	max	0.9474	0.9474	1.0000	1.0000
	count	75.0000	75.0000	633.0000	633.0000

Figures

Distributions

```
In [9]:  df_plot= df[(df.calculated_on=="giant")][['info_total_friend_only','links_supported',
                                                      'density','ave_dist','country','num_node
          s']]
```

```
In [10]: df_plot.groupby('country').num_nodes.describe()
```

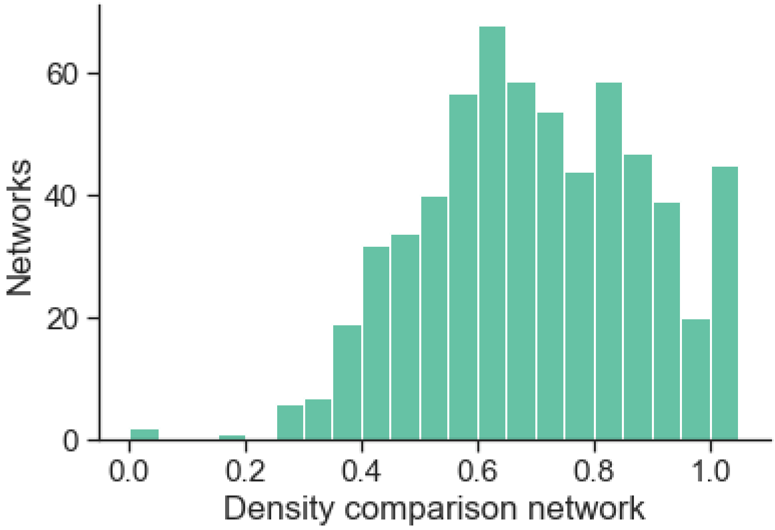
Out[10]:

country	count	mean	std	min	25%	50%	75%	max
India	75.0000	188.8667	57.1485	75.0000	146.0000	175.0000	226.5000	341.0000
Indonesia	633.0000	23.6019	12.3280	2.0000	15.0000	22.0000	31.0000	82.0000

```
In [11]: df_plot[df_plot.info_total_friend_only==1].country.value_counts()

Out[11]: Indonesia    45
         Name: country, dtype: int64
```

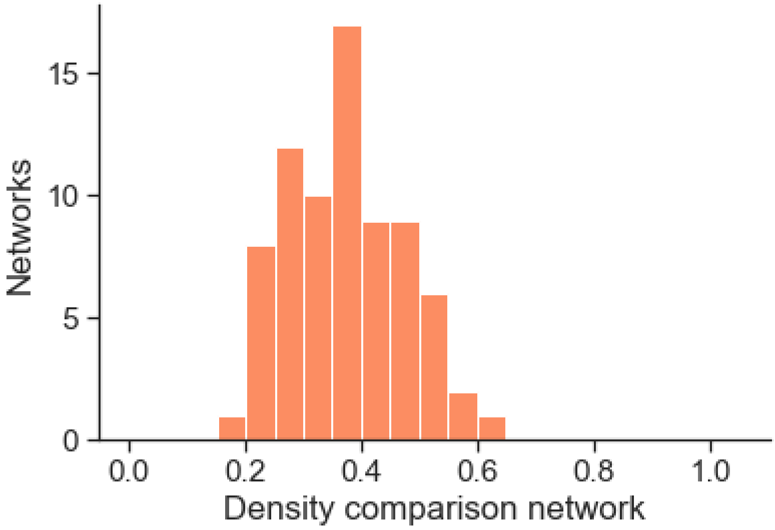
```
In [12]: df_plot[df_plot.links_supported==1].country.value_counts()

Out[12]: Indonesia    127
         Name: country, dtype: int64
```

```
In [13]: sns.distplot(df_plot[df_plot.country=="Indonesia"]['info_total_friend_only'],kde=Fals
         e,bins=np.arange(0,1.06,0.05),
                      color=colors['Indonesia'],hist_kws={'alpha':1})
         plt.ylabel('Networks')
         plt.xlabel('Density comparison network')
         sns.despine()
         plt.savefig('figures/hist_density_comp_indonesia.pdf', bbox_inches='tight')
```



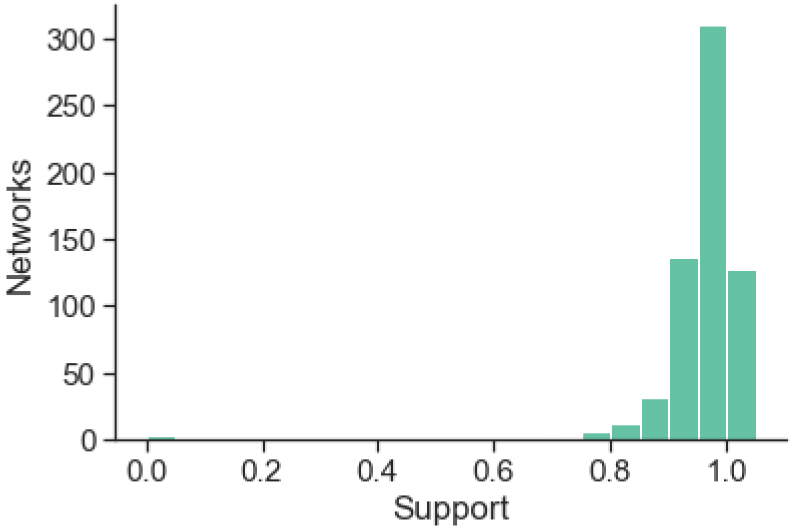
In [14]:

```
sns.distplot(df_plot[df_plot.country=="India"]['info_total_friend_only'],kde=False,bins=np.arange(0,1.06,0.05),
             color=colors['India'],hist_kws={'alpha':1})
plt.ylabel('Networks')
plt.xlabel('Density comparison network')
sns.despine()
plt.savefig('figures/hist_density_comp_india.pdf', bbox_inches='tight')
```



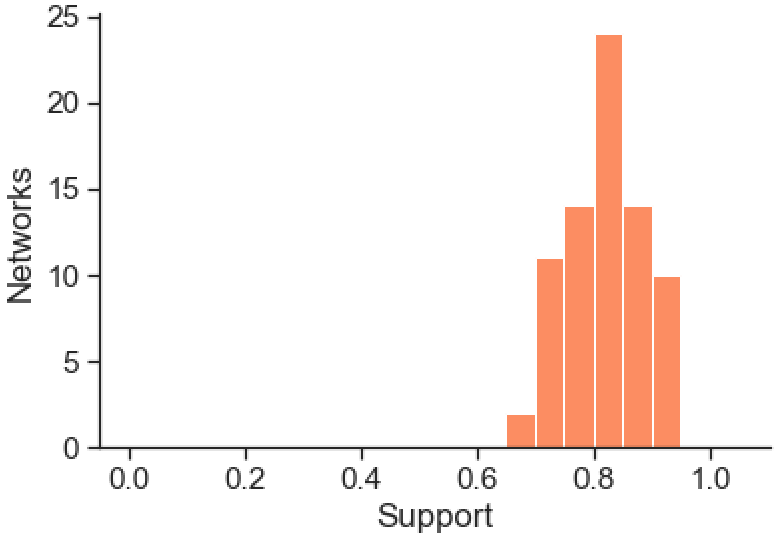
In [15]:

```
sns.distplot(df_plot[df_plot.country=="Indonesia"]['links_supported'],kde=False,bins=np.arange(0,1.06,0.05),
             color=colors['Indonesia'],hist_kws={'alpha':1})
plt.ylabel('Networks')
plt.xlabel('Support')
sns.despine()
plt.savefig('figures/hist_support_indonesia.pdf', bbox_inches='tight')
```



In [16]:

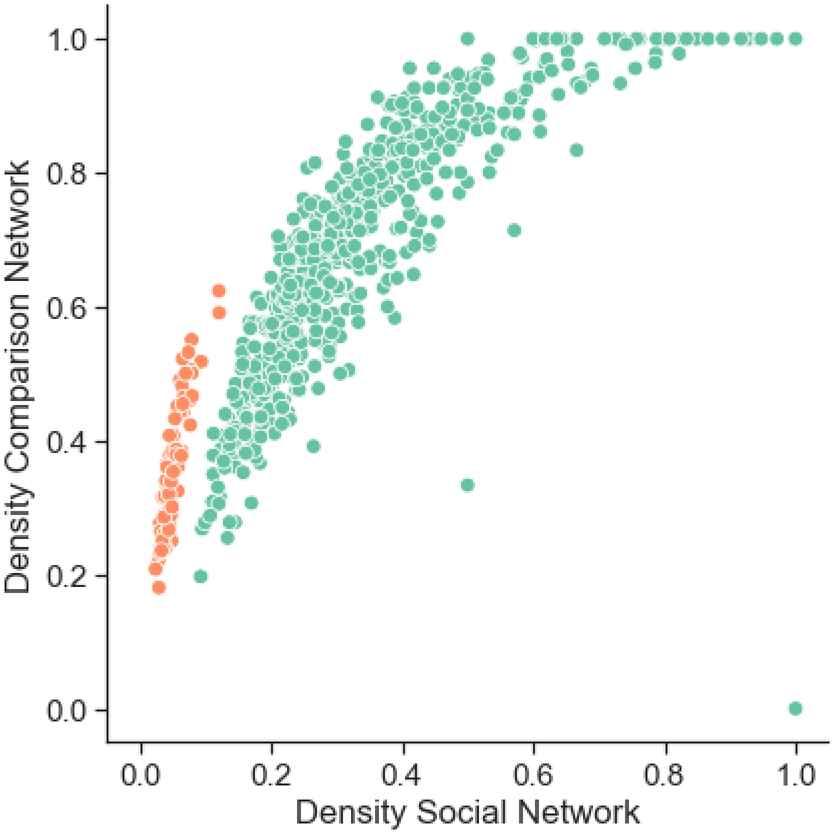
```
sns.distplot(df_plot[df_plot.country=="India"]['links_supported'],kde=False,bins=np.a
range(0,1.06,0.05),
             color=colors['India'],hist_kws={'alpha':1})
plt.ylabel('Networks')
plt.xlabel('Support')
sns.despine()
plt.savefig('figures/hist_support_india.pdf', bbox_inches='tight')
```



Density comparsion network vs social network

In [17]:

```
sns.relplot(x="density", y="info_total_friend_only",height=6,
            hue="country", data=df_plot,palette=colors,aspect=1, s=60,legend=False)
plt.xticks([0,0.2,0.4,0.6,0.8,1])
plt.yticks([0,0.2,0.4,0.6,0.8,1])
plt.ylim(bottom=-0.05,top=1.05)
plt.xlim(left=-0.05,right=1.05)
plt.ylabel('Density Comparison Network')
plt.xlabel('Density Social Network')
plt.savefig('figures/density.pdf', bbox_inches='tight');
```



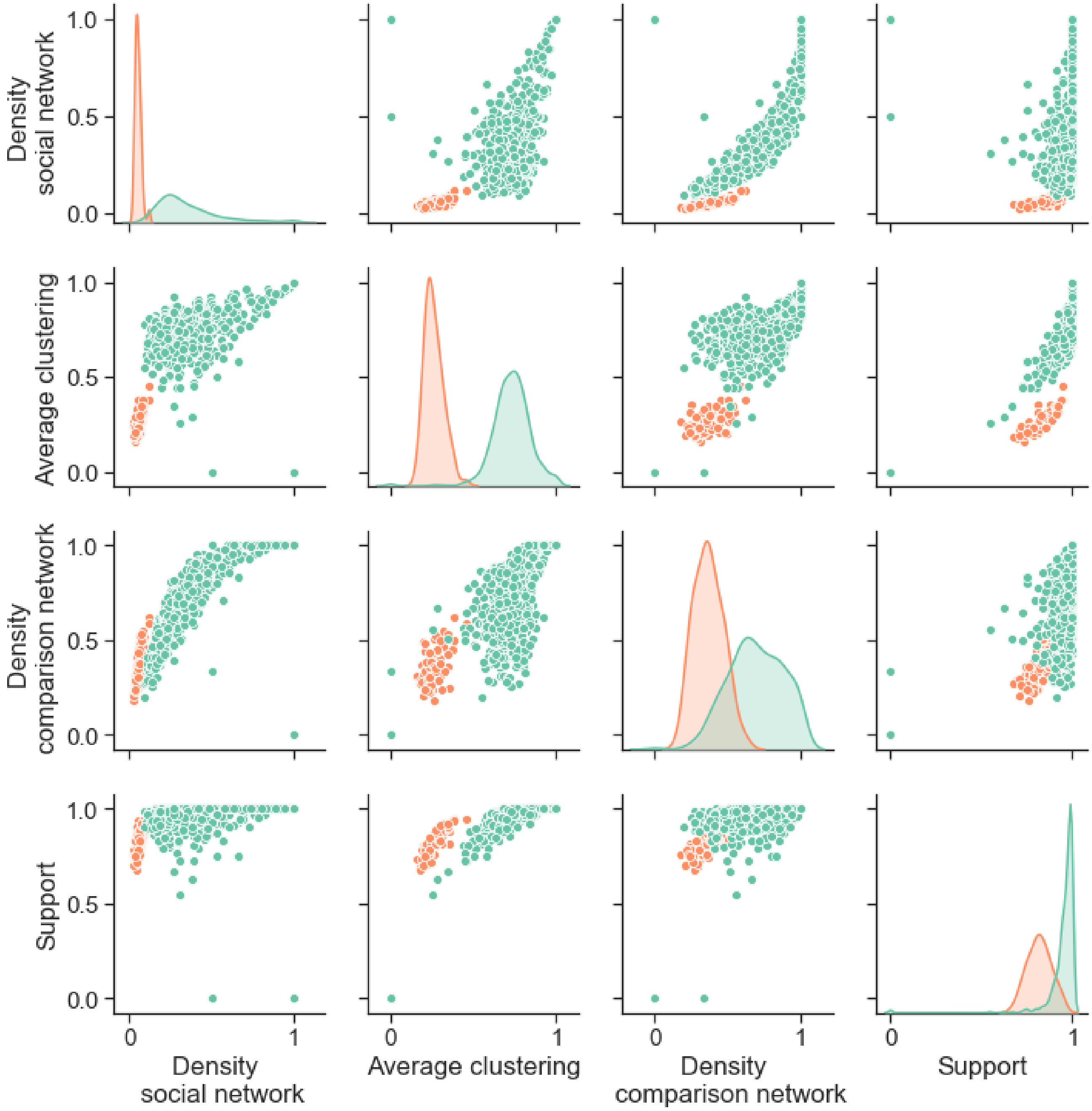
Pairplot

In [18]:

```
df_pairplot = df[df.calculated_on=="giant"][['info_total_friend_only','links_supporte
d','ave_clust','density','country']]
df_pairplot.rename(columns={
                          'density': 'Density \n social network',
                          'ave_clust': 'Average clustering',
                          'info_total_friend_only': 'Density \n comparison network',
                          'links_supported':'Support',
                          'country': 'Country'}, inplace=True)
```

In [19]:

```
ax = sns.pairplot(data=df_pairplot[['Density \n social network','Average clustering',
                                    'Density \n comparison network','Support','Countr
y']],
                   hue='Country',
                   palette=colors)
ax._legend.remove()
ax.savefig('figures/pairplot.pdf');
```



Comparison of mechanisms

In [20]:

```
df_comp = df[(df.info_SP.notnull())&
          (df.country=='Indonesia')&
          (df.calculated_on=='giant')&
          (df.num_nodes<=20)&
          (df.info_expostIC<1)&
          (df.info_total_friend_only>df.info_expostIC)][['key',
                                                          'num_nodes',
                                                          'info_total_friend_only',
                                                          'info_expostIC',
                                                          'comp_supp',
                                                          'info_SP']]
```

In [21]:

```
df_comp['share_partition'] =  df_comp.info_SP/df_comp.info_total_friend_only
```

In [22]:

```
df_comp['share_supp'] =  df_comp.info_expostIC/df_comp.info_total_friend_only
```

In [23]:

```
df_comp.describe()
```

Out[23]:

	num_nodes	info_total_friend_only	info_expostIC	comp_supp	info_SP	share_partition	share_su
count	213.0000	213.0000	213.0000	213.0000	213.0000	213.0000	213.00
mean	14.3803	0.7499	0.6006	0.3769	0.4994	0.6605	0.79
std	3.9989	0.1449	0.1599	0.1321	0.1279	0.0692	0.14
min	4.0000	0.3333	0.0000	0.0000	0.1667	0.5000	0.00
25%	11.0000	0.6667	0.4967	0.2857	0.4211	0.6102	0.71
50%	15.0000	0.7727	0.6222	0.3660	0.5000	0.6512	0.83
75%	18.0000	0.8611	0.7143	0.4505	0.5810	0.7083	0.90
max	20.0000	0.9778	0.8889	0.8000	0.8000	0.8403	0.98

In [24]:

```
df_comp[df_comp.info_expostIC>df_comp.info_SP].key.count()
```

Out[24]:

```
162
```

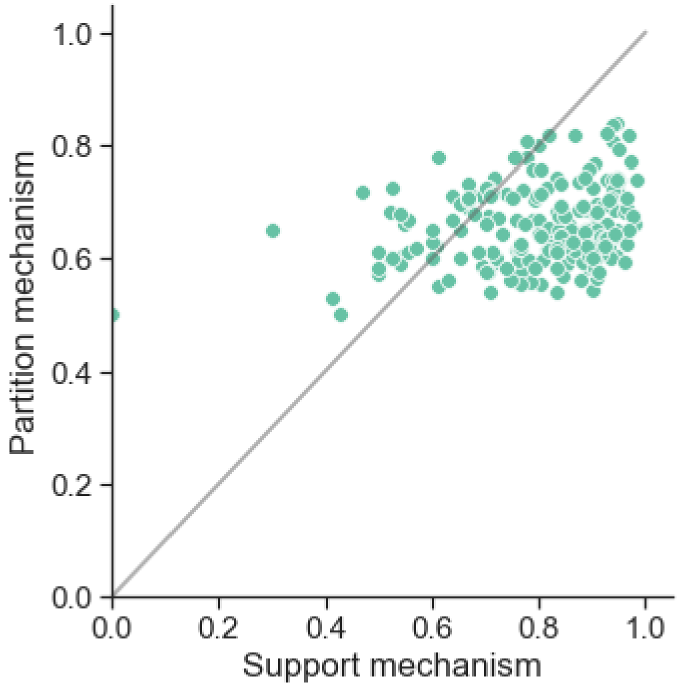
In [25]:

```
df_comp[df_comp.info_expostIC==df_comp.info_SP].key.count()
```

Out[25]:

```
13
```

In [26]:
```
sns.relplot(x="share_supp", y="share_partition",height=5, data=df_comp,
            color=colors["Indonesia"],aspect=1, s=60,legend=False)
plt.plot([0, 1], [0, 1], color = grey_dark, linewidth = 2, alpha=0.5)
plt.xticks([0,0.2,0.4,0.6,0.8,1])
plt.yticks([0,0.2,0.4,0.6,0.8,1])
plt.ylim(bottom=0,top=1.05)
plt.xlim(left=0,right=1.05)
plt.ylabel('Partition mechanism')
plt.xlabel('Support mechanism')
plt.savefig('figures/mechanisms.pdf', bbox_inches='tight');
```



In [27]:
```
df_comp[['info_expostIC','info_SP']].describe()
```

Out[27]:

	info_expostIC	info_SP
count	213.0000	213.0000
mean	0.6006	0.4994
std	0.1599	0.1279
min	0.0000	0.1667
25%	0.4967	0.4211
50%	0.6222	0.5000
75%	0.7143	0.5810
max	0.8889	0.8000