import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

#%matplotlib inline

import networkx as nx

dt=pd.read\_excel('eng\_rat.xlsx') # for rat test with words in G

print(dt)

dt.head(10)

dt[2:5]

dt.iloc[1:5, 0:4]

dt[['Word1', 'Word2', 'Word3','Answer','15s\_acc']].head()

dt['15s\_acc'].hist(bins = 30)

plt.xlabel(' accuracy 15 sec', fontsize=14)

dt['15s\_st'].hist(bins = 30)

plt.xlabel(' accuracy 15 sec', fontsize=14)

plt.scatter(dt['15s\_acc'],dt['15s\_st'])

plt.xlabel(' accuracy 15 sec', fontsize=14)

plt.ylabel(' answer time', fontsize=14)

y=[]

for row in dt.itertuples(): # Iterate over DataFrame

    acc=row.\_9

    if(acc<=32.0): y.append(0)

    else:

        if(acc<=64.0): y.append(1)

        else: y.append(2)

dt['Comp']=y

dt.head()

acc,err\_acc=[],[]

mst,err\_st=[],[]

for c in [0,1,2]:

    t=dt[(dt['Comp']==c)]['15s\_acc'].mean()

    st=dt[(dt['Comp']==c)]['15s\_acc'].std()

    acc.append(t)

    err\_acc.append(st)

    t=dt[(dt['Comp']==c)]['15s\_st'].mean()

    st=dt[(dt['Comp']==c)]['15s\_st'].std()

    mst.append(t)

    err\_st.append(st)

fig, ax = plt.subplots(nrows=1, sharey=False,ncols=2, figsize=(12, 5))

x\_pos = np.arange(3)

ax[0].bar(x\_pos, acc, yerr=err\_acc, align='center', alpha=0.5, ecolor='black', capsize=10)

ax[0].set\_xlabel('complexity', fontsize=14)

ax[0].set\_ylabel('accuracy', fontsize=14)

ax[1].bar(x\_pos, mst, yerr=err\_st, align='center', alpha=0.5, ecolor='black', capsize=10)

ax[1].set\_xlabel('complexity', fontsize=14)

ax[1].set\_ylabel('answer time', fontsize=14)

net=pd.read\_excel('assoc\_eng2.xlsx')

G0=nx.from\_pandas\_edgelist(net, 'source', 'target', edge\_attr=True, create\_using=nx.DiGraph())

print('Nodes:',G0.number\_of\_nodes())

print('Edges:',G0.number\_of\_edges())

print('Density:',nx.density(G0))

print('Transitivity', nx.transitivity(G0))

print(nx.is\_strongly\_connected(G0))

print(len(max(nx.strongly\_connected\_components(G0), key=len)))

H=G0.to\_undirected()

print('Nodes:',H.number\_of\_nodes())

print('Edges:',H.number\_of\_edges())

print('Density:',nx.density(H))

print('Transitivity', nx.transitivity(H))

print(nx.is\_connected(H))

print(len(max(nx.connected\_components(H), key=len)))

plt.show()