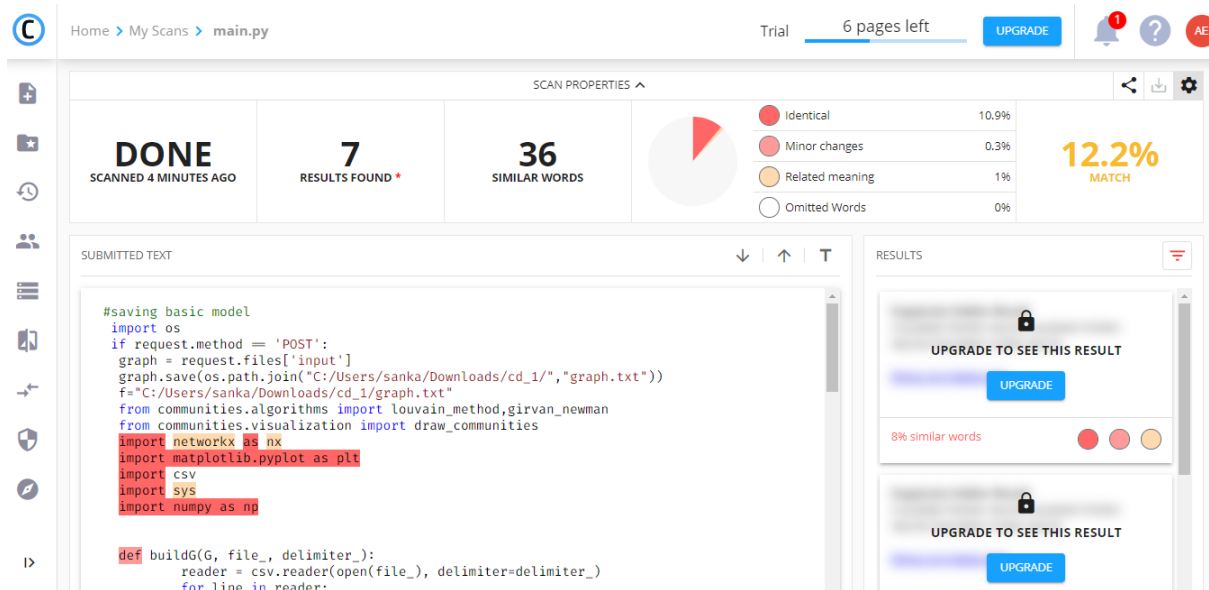


SOURCE CODE PLAGIARISM REPORT – COPYLEAKS.COM



```
#saving basic model
import os
if request.method == 'POST':
    graph = request.files['input']

graph.save(os.path.join("C:/Users/sanka/Downloads/cd_1/
", "graph.txt"))
    f="C:/Users/sanka/Downloads/cd_1/graph.txt"
    from communities.algorithms import
louvain_method,girvan_newman
    from communities.visualization import
draw_communities
    import networkx as nx
    import matplotlib.pyplot as plt
    import csv
    import sys
    import numpy as np

    def buildG(G, file_, delimiter_):
        reader = csv.reader(open(file_),
delimiter=delimiter_)
        for line in reader:

G.add_edge(int(line[0]),int(line[1]))
    G = nx.Graph()
    print(G)
    buildG(G, f, ',')
    #matrix to array
```

```

S= np.array(nx.to_numpy_matrix(G,dtype=int))
print(S)

#louvain
communities, _ = louvain_method(S)
draw_communities(S, communities, False, 'C:/My Web
Sites/dashboard_CD_SENA/community/templates/admin/main/
source/static/louvain.png')
print("communities")
plt.clf()
plt.cla()
plt.close()

#girvan
import community as girvan_newman
import matplotlib.cm as cm
L=G
partition = girvan_newman.best_partition(L)
#Highest_degree
com = set(partition.values())
c_dict = {c: [l for l,i in partition.items() if
i==c ] for c in com}
highest_degree = {l: max(i, key=lambda
x:G.degree(x)) for l,i in c_dict.items()}
a = []
for i in range(0,len(highest_degree)):
    a.append([])
    for j in range(0,2):
        a[i].append([])

for i in range(0,len(highest_degree)):
    a[i][0]=(highest_degree[i])
    a[i][1]=(G.degree(highest_degree[i]))
length=[]
for i in range(0,len(highest_degree)):
    length.append(i)

# draw the graph
pos = nx.spring_layout(L)
cmap = cm.get_cmap('viridis',
max(partition.values()) + 1)
nx.draw_networkx_nodes(L, pos, partition.keys(),
node_size=40,
cmap=cmap, node_color=list(partition.values()))
nx.draw_networkx_edges(L, pos, alpha=0.5)

```

```

plt.savefig('C:/My Web
Sites/dashboard_CD_SENA/community/templates/admin/main/
source/static/girvan.png')
plt.clf()
plt.cla()
plt.close()

f=[]
i=0
y=0
#total no of communities
for i in partition:
    if partition[i] not in f:
        f.append(partition[i])
        y=y+1
#print("Total No of Communities: ", y)
#each community size
import numpy as np
l = np.zeros((y,), dtype=int)
for i in partition:
    k = partition[i]
    l[k]=l[k]+1

for i in range(0,y):
    #print("Community ", i+1," Size: ")
    print(l[i])

#elements in each community
k=len(partition)#no of partitions
g=[]
for i in range(0,y):
    g.append(i)#list of communities

m=[]#community stored as list of lists
for x in range(0,y):
    print("Community: ",x+1)
    q=[]
    for j in partition.keys():
        if(partition[j]==x):
            q.append(j)
    m.append(q)

#generate bar graph
height = 1
bars = range(len(l))

```

```
y_pos = np.arange(len(bars))

# Create bars
plt.bar(y_pos, height)

# Create names on the x-axis
plt.xticks(y_pos, bars)

plt.xlabel('Communities')
plt.ylabel('No. of nodes')
plt.savefig('C:/My Web
Sites/dashboard_CD_SENA/community/templates/admin/main/
source/static/bargraph.png')
plt.clf()
plt.cla()
plt.close()
```