cluster-analysis_doubles

March 12, 2024

[1]:
```
import json
import pandas as pd
import os
from collections import OrderedDict
import math
```

- Loading the dataset

[2]:
```
df = pd.read_csv("cluster_doubles.csv")

#String of nodes to list of nodes
for ind,row in df.iterrows():
    for node in row["nodes"]:
        strr = row["nodes"].strip('{').strip('}')
        strr = strr.strip("'")
        lst = strr.split(',')
        for i in range(0,len(lst)):
            lst[i]=lst[i].strip(" ").strip("'")
            # lst[i] = int(lst[i])
    df.at[ind,"nodes"]=lst

df = df.sort_values(by = 'total_nodes',ascending = False)

with open('player_matches_doubles.json') as f:
    match_counts = json.load(f)
```

For analysis we have considered one match as 6 different matches(6 edges between 4 nodes)

- Matches vs Clusters

[3]:
```
def print_cluster_percentage(matches_percent, threshold, clusters_till_now,␣
 ↪total_clusters):
    cluster_percent = clusters_till_now / total_clusters
    print(f"Around {threshold}% of the matches are played by {100 *␣
 ↪cluster_percent}% clusters")



thresholds = [100, 70, 50, 30] #Input thresholds
total_matches = df.sum(axis = 0, skipna = True)['total_matches']
```

```
total_clusters = len(df)
print("Total Matches: ",total_matches)
print("Actual Total Matches: ",total_matches/6)
print("Total clusters: ",total_clusters)
matches_till_now = 0
clusters_till_now = 0
flags = OrderedDict.fromkeys(thresholds, True)

for _, row in df.iterrows():
    clusters_till_now += 1
    matches_till_now += row['total_matches']
    matches_percent = 100 * matches_till_now / total_matches

    for threshold in sorted(flags.keys(),reverse= True):
        if matches_percent >= threshold:
            print_cluster_percentage(matches_percent, threshold,␣
 ↪clusters_till_now, total_clusters)
            flags.pop(threshold)
            break
```

```
Total Matches:  73182.0
Actual Total Matches:  12197.0
Total clusters:  403
Around 50% of the matches are played by 0.24813895781637718% clusters
Around 30% of the matches are played by 0.49627791563275436% clusters
Around 70% of the matches are played by 7.444168734491314% clusters
Around 100% of the matches are played by 100.0% clusters
```

- Cluster vs Players

[4]:
```
#Change these conditions accordingly
conditions = [
    ("== 2     ", 2),
    ("== 3     ", 3),
    ("== 4     ", 4),
    ("== 5     ", 5),
    (">5 & <=10", (5, 10)),
    (">10      ", 10)
]

# Print total clusters
print("Total Clusters: ", total_clusters)
print("Unique list of cluster sizes: ",df['total_nodes'].unique())

# print("Unique list with frequency:")    #Uncomment to see frequency count
# print(df['total_nodes'].value_counts())

# Calculate the percentage for each condition
```

```
for condition_label, condition_value in conditions:
    if isinstance(condition_value, int):
        subset = df[df['total_nodes'] == condition_value]
    else:
        subset = df[(df['total_nodes'] > condition_value[0]) &␣
 ↪(df['total_nodes'] <= condition_value[1])]

    percentage = len(subset) / total_clusters * 100
    print(f"Clusters with players {condition_label}: {percentage:.2f}% of the␣
 ↪total clusters or {len(subset)} clusters")
```

```
Total Clusters:  403
Unique list of cluster sizes:  [1307   26   25   23   22   20   18   16   14
13   12   11   10    9
    8    7    6    5    4]
Clusters with players == 2      : 0.00% of the total clusters or 0 clusters
Clusters with players == 3      : 0.00% of the total clusters or 0 clusters
Clusters with players == 4      : 55.83% of the total clusters or 225 clusters
Clusters with players == 5      : 15.14% of the total clusters or 61 clusters
Clusters with players >5 & <=10: 23.33% of the total clusters or 94 clusters
Clusters with players >10       : 1.99% of the total clusters or 8 clusters
```

- Cluster vs Players vs Matches

[5]:
```
# Calculate the average matches played per person for each cluster size
average_matches_per_person = {}
average_matches_per_cluster = {}
for size in range(2, max(df['total_nodes'])+1):
    cluster_size_df = df[df['total_nodes'] == size]
    if(len(df))==0:
        continue
    average_matches_per_person[size] = cluster_size_df['total_matches'].mean() /
 ↪ size
    average_matches_per_cluster[size] = cluster_size_df['total_matches'].mean()



# For cluster size between 5 and 10
cluster_size_df = df[(df['total_nodes'] >= 5) & (df['total_nodes'] <= 10)]
average_matches_per_person['(5,10]'] = cluster_size_df['total_matches'].mean() /
 ↪ cluster_size_df['total_nodes'].mean()
average_matches_per_cluster['(5,10]'] = cluster_size_df['total_matches'].mean()

# For cluster size greater than 10
cluster_size_df = df[df['total_nodes'] > 10]
average_matches_per_person['(10,max]'] = cluster_size_df['total_matches'].
 ↪mean() / cluster_size_df['total_nodes'].mean()
```

```
average_matches_per_cluster['(10,max]'] = cluster_size_df['total_matches'].
 ↪mean()

# Print the results
for size_range, average_matches in average_matches_per_person.items():
    if math.isnan(average_matches):
        continue
    print(f"Actual Avg matches/person in a cluster having size {size_range}:␣
 ↪{4*average_matches/6:.2f}")
    print(f"Actual Avg matches/cluster having size {size_range}:␣
 ↪{average_matches_per_cluster[size_range]/6:.2f}\n")
```

```
Actual Avg matches/person in a cluster having size 4: 4.79
Actual Avg matches/cluster having size 4: 4.79

Actual Avg matches/person in a cluster having size 5: 12.10
Actual Avg matches/cluster having size 5: 15.13

Actual Avg matches/person in a cluster having size 6: 8.77
Actual Avg matches/cluster having size 6: 13.15

Actual Avg matches/person in a cluster having size 7: 12.24
Actual Avg matches/cluster having size 7: 21.42

Actual Avg matches/person in a cluster having size 8: 8.65
Actual Avg matches/cluster having size 8: 17.31

Actual Avg matches/person in a cluster having size 9: 12.28
Actual Avg matches/cluster having size 9: 27.64

Actual Avg matches/person in a cluster having size 10: 10.25
Actual Avg matches/cluster having size 10: 25.62

Actual Avg matches/person in a cluster having size 11: 24.55
Actual Avg matches/cluster having size 11: 67.50

Actual Avg matches/person in a cluster having size 12: 13.17
Actual Avg matches/cluster having size 12: 39.50

Actual Avg matches/person in a cluster having size 13: 18.00
Actual Avg matches/cluster having size 13: 58.50

Actual Avg matches/person in a cluster having size 14: 8.00
Actual Avg matches/cluster having size 14: 28.00

Actual Avg matches/person in a cluster having size 16: 52.00
Actual Avg matches/cluster having size 16: 208.00
Actual Avg matches/person in a cluster having size 18: 10.22
Actual Avg matches/cluster having size 18: 46.00

Actual Avg matches/person in a cluster having size 20: 8.60
Actual Avg matches/cluster having size 20: 43.00

Actual Avg matches/person in a cluster having size 22: 11.27
Actual Avg matches/cluster having size 22: 62.00

Actual Avg matches/person in a cluster having size 23: 149.04
Actual Avg matches/cluster having size 23: 857.00

Actual Avg matches/person in a cluster having size 25: 5.92
Actual Avg matches/cluster having size 25: 37.00

Actual Avg matches/person in a cluster having size 26: 6.15
Actual Avg matches/cluster having size 26: 40.00

Actual Avg matches/person in a cluster having size 1307: 18.71
Actual Avg matches/cluster having size 1307: 6114.00

Actual Avg matches/person in a cluster having size (5,10]: 11.12
Actual Avg matches/cluster having size (5,10]: 17.87

Actual Avg matches/person in a cluster having size (10,max]: 20.02
Actual Avg matches/cluster having size (10,max]: 363.00
```

- Players vs Matches

[6]:
```
arr = []

for key,value in match_counts.items():
    arr.append((value,key))

arr.sort(reverse=True)

def print_cluster_percentage(matches_percent, threshold, players_till_now,␣
 ↪total_players):
    player_percent = players_till_now / total_players
    print(f"Around {threshold}% of the matches are played by {100 *␣
 ↪player_percent}% players")


thresholds = [200, 70, 50, 30] #Input thresholds - Can go upto 200%
total_matches = df.sum(axis = 0, skipna = True)['total_matches']
total_players = len(match_counts)
```

```
print("Actual Total Matches: ",total_matches/6)
print("Total Players: ",total_players)
matches_till_now = 0
players_till_now = 0

flags = OrderedDict.fromkeys(thresholds, True)

for iter in arr:
    players_till_now += 1
    matches_till_now += iter[0]
    matches_percent = 100 * matches_till_now / total_matches

    for threshold in sorted(flags.keys(),reverse= True):
        if matches_percent >= threshold:
            print_cluster_percentage(matches_percent, threshold,␣
↪players_till_now, total_players)
            flags.pop(threshold)
            break
```

```
Actual Total Matches:  12197.0
Total Players:  3564
Around 30% of the matches are played by 0.44893378226711567% players
Around 50% of the matches are played by 1.0662177328843996% players
Around 70% of the matches are played by 2.216610549943883% players
Around 200% of the matches are played by 100.0% players
```

- Players vs Clusters vs Matches

[7]:
```
# Function to calculate percentage of players with match counts exceeding a␣
↪threshold
def calculate_percentage(df_slice, threshold):
    percent_players_played_above_threshold_matches=0
    for _, row in df_slice.iterrows():
        total_matches = row['total_matches']
        total_players = row["total_nodes"]
        players_above_threshold = 0

        for node in row["nodes"]:
            if int(match_counts[str(node)]) >= threshold * total_matches:
                players_above_threshold += 1
                #print(threshold * total_matches,match_counts[str(node)])
        percent_players_played_above_threshold_matches += 100 *␣
↪players_above_threshold/ total_players

    return percent_players_played_above_threshold_matches/len(df_slice)


# INPUTS
```

```python
thresholds = [0,0.02,0.06,0.3] # Change the thresholds accordingly
ranges = [(2,5),(5,8),(8,11),(11,max(df['total_nodes'])+1)] # Change the ranges␣
↪accordingly


cluster_frequency = {}

for lims in ranges:
    cluster_analysis_discrete = {}
    for cluster_size in range(lims[0],lims[1]):
        if len(df[df["total_nodes"]==cluster_size])==0:
            continue

        lst = []
        mean_percentages = {}
        df_slice = df[df["total_nodes"]==cluster_size]
        for threshold in thresholds:
            percentages = []

            cluster_frequency[cluster_size]=len(df_slice)
            mean_percentages[threshold*100] = calculate_percentage(df_slice,␣
↪threshold)

#         print(f"Cluster Size: {cluster_size}")
#         for threshold, percentage in mean_percentages.items():
#             print(f"Percentage of players with more than {threshold}% matches:
↪ {percentage:.2f}%")

        cluster_analysis_discrete[cluster_size] = list(mean_percentages.
↪items()),sum(df_slice['total_nodes'])
#     print(cluster_analysis_discrete) #Ouput Array with weights

    #computing weighted average array from cluster_analysis_discrete
    arr = {}
    total_weight = 0
    for key,value in cluster_analysis_discrete.items():
        total_weight+=value[1]
        for threshold,percentage in value[0]:
            arr[threshold] = arr.get(threshold,0) + percentage*value[1]

    for key in arr:
        arr[key] = arr[key]/total_weight
        print(f"For the cluster-size range {lims[0],lims[1]}: Avg  % of players␣
↪with more than {key}% matches= {arr[key]:.2f}%")
    print("\n")
```

```
For the cluster-size range (2, 5): Avg  % of players with more than 0% matches=
100.00%
For the cluster-size range (2, 5): Avg  % of players with more than 2.0%
matches= 100.00%
For the cluster-size range (2, 5): Avg  % of players with more than 6.0%
matches= 100.00%
For the cluster-size range (2, 5): Avg  % of players with more than 30.0%
matches= 100.00%


For the cluster-size range (5, 8): Avg  % of players with more than 0% matches=
100.00%
For the cluster-size range (5, 8): Avg  % of players with more than 2.0%
matches= 100.00%
For the cluster-size range (5, 8): Avg  % of players with more than 6.0%
matches= 98.46%
For the cluster-size range (5, 8): Avg  % of players with more than 30.0%
matches= 62.27%


For the cluster-size range (8, 11): Avg  % of players with more than 0% matches=
100.00%
For the cluster-size range (8, 11): Avg  % of players with more than 2.0%
matches= 100.00%
For the cluster-size range (8, 11): Avg  % of players with more than 6.0%
matches= 90.46%
For the cluster-size range (8, 11): Avg  % of players with more than 30.0%
matches= 28.98%


For the cluster-size range (11, 1308): Avg  % of players with more than 0%
matches= 100.00%
For the cluster-size range (11, 1308): Avg  % of players with more than 2.0%
matches= 20.32%
For the cluster-size range (11, 1308): Avg  % of players with more than 6.0%
matches= 12.95%
For the cluster-size range (11, 1308): Avg  % of players with more than 30.0%
matches= 2.22%
```

```
[ ]:
```