lab-10

May 29, 2023

#Epidemic model

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

Importing libraries

[ ]:
```
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
import networkx as nx
```

Loading datasdet

[ ]:
```
airport_df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/social network␣
 ↪analysis/LAB10/COVID-19-master/data/airport_df.csv')
connections_df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/social␣
 ↪network analysis/LAB10/COVID-19-master/data/connections_df.csv')
```

[ ]:
```
INFECTED_CITIES = ["Wuhan"]
```

Setting simulation parameters

[ ]:
```
simulation_days = 7
infection_rate = 0.3
```

[ ]:
```
for day in range(simulation_days):
    new_infected_cities = []

    # Step 5: Iterate over infected cities

    for infected_city in INFECTED_CITIES:

        # Step 6: Get all airports of the infected city
```

```
        infected_airports = airport_df[airport_df["City"] ==␣
↪infected_city]["IATA"].values

        # Step 7: Get all connections for the infected city airports

        connections = connections_df[connections_df["Source Airport"].
↪isin(infected_airports)]

        # Step 8: Iterate over susceptible cities in connections

        for susceptible_city in connections["Dest Airport"]:

            # Step 9: Calculate probability of infection

            if np.random.random() < infection_rate:

                # Step 10: If susceptible city is infected, update␣
↪INFECTED_CITIES

                if susceptible_city not in INFECTED_CITIES:
                    new_infected_cities.append(susceptible_city)

    # Step 11: Update INFECTED_CITIES with newly infected cities

    INFECTED_CITIES.extend(new_infected_cities)

    # Step 12: Print infected cities for the current day

    print(f"Day {day + 1}: Infected cities: {', '.join(INFECTED_CITIES)}")
```

```
Day 1: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT
Day 2: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT, NTG, PUS, WUX, FOC, KWL, PVG, SYX, TYN, XIY, DSN,
URC, XIY, ENH, PVG, URC, CIH, PVG, SZX, TYN
Day 3: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT, NTG, PUS, WUX, FOC, KWL, PVG, SYX, TYN, XIY, DSN,
URC, XIY, ENH, PVG, URC, CIH, PVG, SZX, TYN, CGQ, HKG, BAV, NGB, ZUH, YTY, AQG,
KHN, LZH, NGB
Day 4: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
```

```
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT, NTG, PUS, WUX, FOC, KWL, PVG, SYX, TYN, XIY, DSN,
URC, XIY, ENH, PVG, URC, CIH, PVG, SZX, TYN, CGQ, HKG, BAV, NGB, ZUH, YTY, AQG,
KHN, LZH, NGB, DLC, YNZ, LYI, XFN
Day 5: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT, NTG, PUS, WUX, FOC, KWL, PVG, SYX, TYN, XIY, DSN,
URC, XIY, ENH, PVG, URC, CIH, PVG, SZX, TYN, CGQ, HKG, BAV, NGB, ZUH, YTY, AQG,
KHN, LZH, NGB, DLC, YNZ, LYI, XFN
Day 6: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT, NTG, PUS, WUX, FOC, KWL, PVG, SYX, TYN, XIY, DSN,
URC, XIY, ENH, PVG, URC, CIH, PVG, SZX, TYN, CGQ, HKG, BAV, NGB, ZUH, YTY, AQG,
KHN, LZH, NGB, DLC, YNZ, LYI, XFN, HRB
Day 7: Infected cities: Wuhan, CKG, JJN, LJG, CAN, CDG, CNX, CTU, HET, INC, KWE,
SHE, XNN, TPE, BKK, HGH, ICN, KMG, LHW, NNG, PEK, SWA, TAO, TSN, TSA, LJG, HAK,
HET, ICN, CKG, CTU, INC, HGH, KHH, LHW, SHA, SIN, XMN, YCU, WNZ, MFM, CKG, KMG,
TNA, YNT, SIN, PEK, SHE, YNT, NTG, PUS, WUX, FOC, KWL, PVG, SYX, TYN, XIY, DSN,
URC, XIY, ENH, PVG, URC, CIH, PVG, SZX, TYN, CGQ, HKG, BAV, NGB, ZUH, YTY, AQG,
KHN, LZH, NGB, DLC, YNZ, LYI, XFN, HRB, HYN, DMK, LYG, ZYI
```

[ ]:
```python
G = nx.from_pandas_edgelist(connections_df, 'Source Airport', 'Dest Airport',␣
↪create_using=nx.DiGraph())
```

[ ]:
```python
def calculate_infection_probability(susceptible_city):
    # Calculating the infection probability based on some criteria
    # For example, you can use the number of connections to infected cities
    num_connections = G.degree[susceptible_city]
    return num_connections * 0.1  # Adjust the weight as needed
```

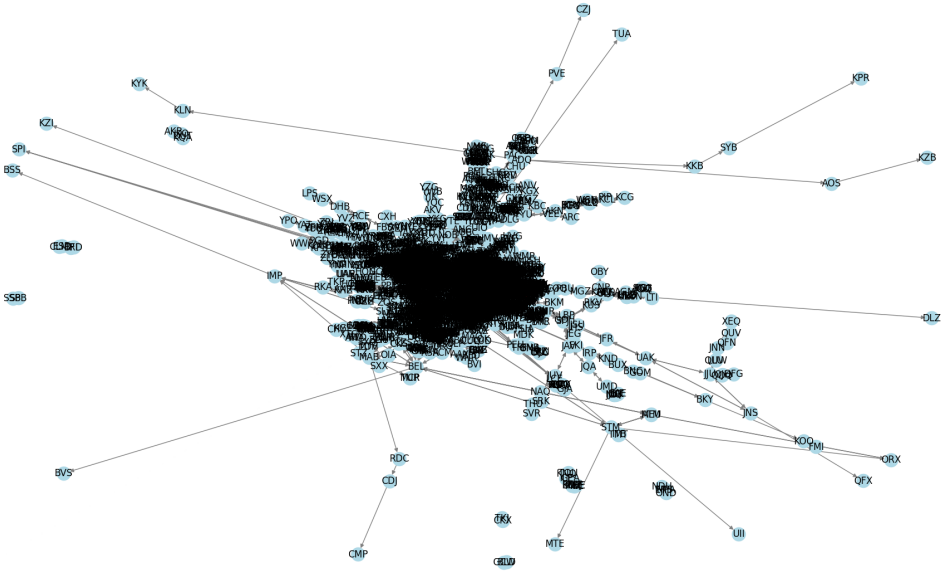
[ ]:
```python
simulation_days = 5  # Change this as needed

for day in range(simulation_days):
    new_infected_cities = []

    for infected_city in INFECTED_CITIES:
        # Get all airports of the infected city
        airports = airport_df[airport_df['City'] == infected_city]['IATA'].
↪tolist()

        for airport in airports:
            # Get all connections for the airport
            connections = G.successors(airport)

            for susceptible_city in connections:
```

```
                # Calculate the probability of infection for the susceptible␣
 ↪city
                probability = calculate_infection_probability(susceptible_city)

                if probability > 0 and susceptible_city not in INFECTED_CITIES:
                    # Check if the susceptible city gets infected
                    if probability > 0.5:  # Adjust the threshold as needed
                        new_infected_cities.append(susceptible_city)

    INFECTED_CITIES.extend(new_infected_cities)
```

[ ]:
```
print("Infected Cities:")
print(INFECTED_CITIES)
```

```
Infected Cities:
['Wuhan', 'CKG', 'JJN', 'LJG', 'CAN', 'CDG', 'CNX', 'CTU', 'HET', 'INC', 'KWE',
'SHE', 'XNN', 'TPE', 'BKK', 'HGH', 'ICN', 'KMG', 'LHW', 'NNG', 'PEK', 'SWA',
'TAO', 'TSN', 'TSA', 'LJG', 'HAK', 'HET', 'ICN', 'CKG', 'CTU', 'INC', 'HGH',
'KHH', 'LHW', 'SHA', 'SIN', 'XMN', 'YCU', 'WNZ', 'MFM', 'CKG', 'KMG', 'TNA',
'YNT', 'SIN', 'PEK', 'SHE', 'YNT', 'NTG', 'PUS', 'WUX', 'FOC', 'KWL', 'PVG',
'SYX', 'TYN', 'XIY', 'DSN', 'URC', 'XIY', 'ENH', 'PVG', 'URC', 'CIH', 'PVG',
'SZX', 'TYN', 'CGQ', 'HKG', 'BAV', 'NGB', 'ZUH', 'YTY', 'AQG', 'KHN', 'LZH',
'NGB', 'DLC', 'YNZ', 'LYI', 'XFN', 'HRB', 'HYN', 'DMK', 'LYG', 'ZYI']
```

[ ]:
```
infected_graph = G.subgraph(INFECTED_CITIES)

pos = nx.spring_layout(G)
plt.figure(figsize=(20, 12))
nx.draw(G, pos, node_color='lightblue', edge_color='gray', with_labels=True)
nx.draw(infected_graph, pos, node_color='red', edge_color='gray',␣
 ↪with_labels=True)
plt.title('Airport Connections with Infected Cities')
plt.show()
```


Airport Connections with Infected Cities
CZJ
TUA
PVE
KPR
KYK
KLN
KZI
SPI
BSS
SYB
KZB
KKB
AOS
LPS
IMP
SSBB
DLZ
XEQ
QUV
QFN
BKY
JNS
KOO
FMI
ORX
QFX
BVS
RDC
CDJ
CMP
MTE
UII