Bu kısımda ise oluşturulan 2. Kod bulunmaktadır. Söz konusu kod kesişim bölgelerindeki enerjileri ayrı ayrı hesaplamamıza ve daha ayrıntılı yorum yapabilmemize olanak sağlamaktadır.

import pandas as pd

import numpy as np

from collections import defaultdict

from numpy import linalg as LA

import hypernetx as hnx

def hyper\_matrix(H):

    n = len(H.nodes)

    matrix = np.zeros((n, n))

    grouped = defaultdict(set)

    for edge\_id in H.edges():

        if edge\_id in H.incidence\_dict:

            nodes\_in\_edge = H.incidence\_dict[edge\_id]

            for node in nodes\_in\_edge:

                grouped[node].add(edge\_id)

    nodes = list(H.nodes)

    for i in range(n):

        for j in range(i + 1, n):

            node\_i\_name = nodes[i]

            node\_j\_name = nodes[j]

            if node\_i\_name in grouped and node\_j\_name in grouped:

                common\_edges = len(grouped[node\_i\_name] & grouped[node\_j\_name])

                matrix[i][j] = common\_edges

                matrix[j][i] = common\_edges

    return matrix

def hyper\_energy(H):

    if len(H.nodes) == 0:

        return 0.0

    m = hyper\_matrix(H)

    e = np.abs(LA.eigvalsh(m))

    return np.sum(e)

data = pd.read\_excel(r"/content/TakımYıldızları (1).xlsx")

Cluster9060 = []

Cluster6030 = []

Cluster3000 = []

Cluster0030 = []

Cluster3060 = []

Cluster6090 = []

for name, enlem\_str in zip(data["NAME"], data["ENLEM"]):

    try:

        degree\_part\_str = enlem\_str.split(':')[0]

        value = int(float(degree\_part\_str))

    except (ValueError, IndexError) as e:

        continue

    if 60 <= value <= 90:

        Cluster6090.append([name, value])

    elif 30 <= value < 60:

        Cluster3060.append([name, value])

    elif 0 <= value < 30:

        Cluster0030.append([name, value])

    elif -30 <= value < 0:

        Cluster3000.append([name, value])

    elif -60 <= value < -30:

        Cluster6030.append([name, value])

    elif -90 <= value < -60:

        Cluster9060.append([name, value])

Tolerans = 1

e1\_nodes = []

for u in Cluster9060:

    if u[1] in range(-60 - Tolerans, -60):

        e1\_nodes.append(u)

for v in Cluster6030:

    if v[1] in range(-59, -59 + Tolerans):

        e1\_nodes.append(v)

e2\_nodes = []

for u in Cluster6030:

    if u[1] in range(-30 - Tolerans, -30):

        e2\_nodes.append(u)

for v in Cluster3000:

    if v[1] in range(-29, -29 + Tolerans):

        e2\_nodes.append(v)

e3\_nodes = []

for u in Cluster3000:

    if u[1] in range(-Tolerans, 0):

        e3\_nodes.append(u)

for v in Cluster0030:

    if v[1] in range(1, 1 + Tolerans):

        e3\_nodes.append(v)

e4\_nodes = []

for u in Cluster0030:

    if u[1] in range(30 - Tolerans, 30):

        e4\_nodes.append(u)

for v in Cluster3060:

    if v[1] in range(31, 31 + Tolerans):

        e4\_nodes.append(v)

e5\_nodes = []

for u in Cluster3060:

    if u[1] in range(60 - Tolerans, 60):

        e5\_nodes.append(u)

for v in Cluster6090:

    if v[1] in range(61, 61 + Tolerans):

        e5\_nodes.append(v)

all\_edges\_dict = {

    "Cluster9060": [node[0] for node in Cluster9060],

    "Cluster6030": [node[0] for node in Cluster6030],

    "Cluster3000": [node[0] for node in Cluster3000],

    "Cluster0030": [node[0] for node in Cluster0030],

    "Cluster3060": [node[0] for node in Cluster3060],

    "Cluster6090": [node[0] for node in Cluster6090]

}

if e1\_nodes:

    all\_edges\_dict["e1"] = [node[0] for node in e1\_nodes]

if e2\_nodes:

    all\_edges\_dict["e2"] = [node[0] for node in e2\_nodes]

if e3\_nodes:

    all\_edges\_dict["e3"] = [node[0] for node in e3\_nodes]

if e4\_nodes:

    all\_edges\_dict["e4"] = [node[0] for node in e4\_nodes]

if e5\_nodes:

    all\_edges\_dict["e5"] = [node[0] for node in e5\_nodes]

H\_full = hnx.Hypergraph(all\_edges\_dict)

full\_hypergraph\_energy = hyper\_energy(H\_full)

print(f"\n--- Tüm Hipergrafın Toplam Enerjisi ---")

print(f"Ana hipergrafın (tüm kümeler ve kesişim bölgeleri dahil) enerjisi: {full\_hypergraph\_energy:.4f}")

print("---------------------------------------")

print("\n--- Bireysel Kesişim Bölgelerinin Hiper Enerjileri (Tekrar) ---")

if e1\_nodes:

    df\_e1 = pd.DataFrame({"Edges": ["e1"] \* len(e1\_nodes), "Nodes": [node[0] for node in e1\_nodes]})

    H\_e1 = hnx.Hypergraph(df\_e1, edge\_col="Edges", node\_col="Nodes")

    energy\_e1 = hyper\_energy(H\_e1)

    print(f"e1 kesişim bölgesinin enerjisi: {energy\_e1:.4f}")

else:

    print("e1 kesişim bölgesi boş, enerji hesaplanamadı.")

if e2\_nodes:

    df\_e2 = pd.DataFrame({"Edges": ["e2"] \* len(e2\_nodes), "Nodes": [node[0] for node in e2\_nodes]})

    H\_e2 = hnx.Hypergraph(df\_e2, edge\_col="Edges", node\_col="Nodes")

    energy\_e2 = hyper\_energy(H\_e2)

    print(f"e2 kesişim bölgesinin enerjisi: {energy\_e2:.4f}")

else:

    print("e2 kesişim bölgesi boş, enerji hesaplanamadı.")

if e3\_nodes:

    df\_e3 = pd.DataFrame({"Edges": ["e3"] \* len(e3\_nodes), "Nodes": [node[0] for node in e3\_nodes]})

    H\_e3 = hnx.Hypergraph(df\_e3, edge\_col="Edges", node\_col="Nodes")

    energy\_e3 = hyper\_energy(H\_e3)

    print(f"e3 kesişim bölgesinin enerjisi: {energy\_e3:.4f}")

else:

    print("e3 kesişim bölgesi boş, enerji hesaplanamadı.")

if e4\_nodes:

    df\_e4 = pd.DataFrame({"Edges": ["e4"] \* len(e4\_nodes), "Nodes": [node[0] for node in e4\_nodes]})

    H\_e4 = hnx.Hypergraph(df\_e4, edge\_col="Edges", node\_col="Nodes")

    energy\_e4 = hyper\_energy(H\_e4)

    print(f"e4 kesişim bölgesinin enerjisi: {energy\_e4:.4f}")

else:

    print("e4 kesişim bölgesi boş, enerji hesaplanamadı.")

if e5\_nodes:

    df\_e5 = pd.DataFrame({"Edges": ["e5"] \* len(e5\_nodes), "Nodes": [node[0] for node in e5\_nodes]})

    H\_e5 = hnx.Hypergraph(df\_e5, edge\_col="Edges", node\_col="Nodes")

    energy\_e5 = hyper\_energy(H\_e5)

    print(f"e5 kesişim bölgesinin enerjisi: {energy\_e5:.4f}")

else:

    print("e5 kesişim bölgesi boş, enerji hesaplanamadı.")

print("---------------------------------------------------------")