```
from google.colab import drive
import pandas as pd
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
import matplotlib.pyplot as plt
from itertools import combinations
import seaborn as sns

drive.mount("/content/drive")
```

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

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

```
stop_words = set(stopwords.words("english"))

extra_words = {
    "years", "old", "height", "meter", "kg", "cm", "color", "eye",
"hair",
    "male", "female", "working", "job", "govt", "assistant",
"professor",
    "true", "false", "children", "brown", "black", "white", "pale",
"dark",
    "student", "person", "occupation", "weight", "actor", "leo",
"english",
    "telugu", "japanese", "american", "native", "language", "study"
}

file_path = "/content/drive/MyDrive/Course Work/Sem 4/Data Analysis
and Visualization/Lab 6/Person_Data.xlsx"

df = pd.read_excel(file_path, sheet_name="Sheet1")

df
```

```
{"type":"dataframe","variable_name":"df"}
```

```
data = df.iloc[:, 1:]

data
```

```
{"summary":"{\n  \"name\": \"data\",\n  \"rows\": 49,\n  \"fields\":
[\n    {\n      \"column\": \"Person\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 46,\n
```

```
\"samples\": [\n          \"Sashank\",\n          \"Priyanshu\",\n
\"AK\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Attributes\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 41,\n        \"samples\":
[\n          \"Beautiful\",\n          \"43 Years\",\n          \"6
feet\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 3\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 43,\n        \"samples\":
[\n          \"happy\",\n          \"white skin\",\n
\"beutiful\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 4\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 43,\n        \"samples\":
[\n          \"artistic\",\n          \"big face\",\n          \"good
dressing\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 5\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 43,\n        \"samples\":
[\n          \"brown\",\n          \"lazy\",\n          \"humour\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Unnamed: 6\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 40,\n        \"samples\": [\n
\"mischievious\",\n          \"Chill\",\n          \"Helpful\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Unnamed: 7\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 37,\n        \"samples\": [\n          \"always
panics\",\n          \"Kind\",\n          \"Sportsman\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Unnamed: 8\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 33,\n        \"samples\": [\n
\"leo\",\n          \"big eyes\",\n          \"disciplined\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 9\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 29,\n        \"samples\":
[\n          \"white\",\n          \"girl\",\n          \"Eye Color:
Black\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 10\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 20,\n
\"samples\": [\n          \"Courageous\",\n          \"shy\",\n
\"eye glasses\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 11\",\n      \"properties\": {\n        \"dtype\":
```

\"category\",\n   \"num_unique_values\": 15,\n
\"samples\": [\n   \"strong\",\n   \"mental\",\n
\"Caring\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Unnamed: 12\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 14,\n
\"samples\": [\n   \"motivator\",\n   \"natural\",\n
\"Emotionaally Distant\"\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Unnamed: 13\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 9,\n
\"samples\": [\n   \"inspiring\",\n   \"simple\",\n
\"sport\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Unnamed: 14\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 8,\n   \"samples\":
[\n   \"relational\",\n   \"sketcher\",\n
\"Caring\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Unnamed: 15\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 5,\n   \"samples\":
[\n   \"Hair Type: Silky\",\n   \"friend\",\n
\"foodie\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Unnamed: 16\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   \"cry often\",\n   \"Skin Type: Dry\"\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Unnamed: 17\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n   \"samples\": [\n
\"pretty\",\n   \"Fav Color: Yellow\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Unnamed: 18\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n   \"samples\": [\n   \"Fav
Food: Upma\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Unnamed: 19\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": null,\n   \"min\": null,\n
\"max\": null,\n   \"num_unique_values\": 0,\n
\"samples\": [],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Unnamed: 20\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 1,\n   \"samples\":
[],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"data"}

```
data.drop([16],axis=0,inplace=True)
```

```
data.reset_index(drop=True, inplace=True)

data
```


```
{"summary":"{\n  \"name\": \"data\",\n  \"rows\": 48,\n  \"fields\":
[\n    {\n      \"column\": \"Person\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 46,\n
\"samples\": [\n          \"Sashank\",\n          \"Priyanshu\",\n
\"AK\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Attributes\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 41,\n        \"samples\":
[\n          \"Beautiful\",\n          \"43 Years\",\n          \"6
feet\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 3\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 43,\n        \"samples\":
[\n          \"happy\",\n          \"white skin\",\n
\"beutiful\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 4\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 43,\n        \"samples\":
[\n          \"artistic\",\n          \"big face\",\n          \"good
dressing\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 5\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 43,\n        \"samples\":
[\n          \"brown\",\n          \"lazy\",\n          \"humour\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Unnamed: 6\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 40,\n        \"samples\": [\n
\"mischievious\",\n          \"Chill\",\n          \"Helpful\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Unnamed: 7\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 37,\n        \"samples\": [\n          \"always
panics\",\n          \"Kind\",\n          \"Sportsman\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Unnamed: 8\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 33,\n        \"samples\": [\n
\"leo\",\n          \"big eyes\",\n          \"disciplined\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 9\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 29,\n        \"samples\":
[\n          \"white\",\n          \"girl\",\n          \"Eye Color:
Black\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"Unnamed: 10\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 20,\n
\"samples\": [\n          \"Courageous\",\n          \"shy\",\n
\"eye glasses\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 11\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 15,\n
\"samples\": [\n          \"strong\",\n          \"mental\",\n
\"Caring\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 12\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 14,\n
\"samples\": [\n          \"motivator\",\n          \"natural\",\n
\"Emotionaally Distant\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Unnamed: 13\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 9,\n
\"samples\": [\n          \"inspiring\",\n          \"simple\",\n
\"sport\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 14\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 8,\n        \"samples\":
[\n          \"relational\",\n          \"sketcher\",\n
\"Caring\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 15\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"Hair Type: Silky\",\n          \"friend\",\n
\"foodie\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 16\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"cry often\",\n          \"Skin Type: Dry\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Unnamed: 17\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"pretty\",\n          \"Fav Color: Yellow\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Unnamed: 18\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          \"Fav
Food: Upma\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 19\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": null,\n        \"min\": null,\n
\"max\": null,\n        \"num_unique_values\": 0,\n
\"samples\": [],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"Unnamed: 20\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 1,\n        \"samples\":
[],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"data"}
```

```
def preprocess_attributes(attributes):
    cleaned_attributes = set()
    for attr in attributes:
        if isinstance(attr, str):  # Ensure it's a valid string
            attr = attr.lower().strip()  # Convert to lowercase and
strip spaces
            attr = re.sub(r'[^a-zA-Z ]+', '', attr)  # Remove numbers
and special characters
            words = attr.split()
            filtered_words = [word for word in words if word not in
stop_words and word not in extra_words]
            if filtered_words:
                cleaned_attributes.add(" ".join(filtered_words))  #
Reconstruct phrase

    return cleaned_attributes

def create_person_attribute_dict(df):
    person_dict = {}

    for _, row in df.iterrows():
        person = row.iloc[0]  # First element is the name
        attributes = row.iloc[1:].dropna().tolist()  # Drop NaN values
and get attributes
        person_dict[person] = preprocess_attributes(attributes)

    return person_dict

def jaccard_distance(set1, set2):
    intersection = len(set1 & set2)
    union = len(set1 | set2)
    return 1 - (intersection / union) if union != 0 else 1

def compute_pairwise_distances(person_dict):
  persons = list(person_dict.keys())
  distances = []
  for p1, p2 in combinations(persons, 2):
    distance = jaccard_distance(person_dict[p1], person_dict[p2])
    distances.append(distance)
  return distances

def compute_percentage_of_pairs(thresholds, distances):
  percentages = []
  for threshold in thresholds:
    count = sum(1 for d in distances if d <= threshold)
    percentage = (count / len(distances)) * 100
```

```
    percentages.append(percentage)
  return percentages

def plot_linear_plot(x, y, x_label, y_label, title):
  plt.plot(x, y, marker='o')
  plt.xlabel(x_label)
  plt.ylabel(y_label)
  plt.title(title)
  plt.grid(True)
  plt.show()

person_attribute_dict = create_person_attribute_dict(data)

person_attribute_dict
```

```
{'A': {'ambitions',
  'caring',
  'courageous',
  'emotionaally distant',
  'funny',
  'hard',
  'intellegent',
  'kind',
  'mature',
  'short tempered',
  'understanding'},
 'Dad': {'caring',
  'emotion less',
  'hard',
  'holy',
  'honest',
  'loyal',
  'progressive mind',
  'self confidance',
  'trustwarthy'},
 'Mother': {'beautiful',
  'best cook',
  'caring',
  'emotional',
  'hard',
  'helping',
  'loving',
  'short',
  'short tempered',
  'strong'},
 'Rohit': {'casual',
  'helpful',
  'humorous',
  'low marks',
  'makes fun everyone',
```

```
  'professional',
  'shortcut',
  'standup comedian'},
 'B': {'caring',
  'funny',
  'handsome',
  'honest',
  'intelligent',
  'kind',
  'loyal',
  'rich',
  'romantic',
  'slim',
  'sportsman',
  'voilent',
  'writer'},
 'Pratyush Kumar': {'funny', 'hero', 'short', 'tempered', 'thin'},
 'C': {'agrees everything',
  'compromising',
  'emotions',
  'funny',
  'good memory',
  'helping',
  'reasonable',
  'relational',
  'rich',
  'selfless',
  'simple',
  'smart',
  'supportive',
  'understanding'},
 'D ': {'caring',
  'foodie',
  'helping nature',
  'ignorance',
  'reader',
  'short tempered'},
 'Suresh': {'awareness', 'geedy new thing', 'overcomig failure',
'socialist'},
 'Sreekar': {'fav mov inception',
  'feet',
  'likes cricket',
  'likes movies',
  'reader',
  'skill play keyboard',
  'vijaywada ap'},
 'E': {'arogent',
  'beautiful',
  'disciplined',
```

```
  'early riser',
  'emotional',
  'feet',
  'listen song'},
 'Akash': {'engineering', 'lang telgu', 'slim'},
 'F': {'age',
  'atheletic figure',
  'basketball player',
  'calm composed',
  'feet',
  'highly competetive',
  'sarcastic'},
 'Anil': {'age',
  'feet',
  'food biryani',
  'hobbies football comics',
  'movies songs',
  'skill computer',
  'skin'},
 'B Vijaya Laxmi': {'emp', 'feet', 'kind', 'loving'},
 'H': {'asstt',
  'bindi',
  'earring',
  'fav food upma',
  'fav yellow',
  'glasses',
  'h',
  'married',
  'skin type dry',
  'skintone',
  'type silky'},
 'AA': {'arrogant', 'helpful', 'hyper', 'overconfident', 'short
temper'},
 'AB': {'bad sports',
  'chill',
  'communication',
  'effortless',
  'highly intelligent',
  'intriguing',
  'lazy'},
 'MANOPRIYA': {'caring',
  'childish',
  'creative',
  'cute',
  'handwriting',
  'intelligent',
  'pretty eyes',
  'thoughtful'},
 'AD': {'always panics',
```

```
  'big eyes',
  'caring',
  'curly',
  'emotional',
  'friendly',
  'funny',
  'hardworking',
  'loving',
  'multitasking',
  'shorter',
  'smart'},
 'AE': {'caring',
  'coward',
  'cry often',
  'cute',
  'foodie',
  'girl',
  'happy',
  'intelligent',
  'loving',
  'mischievious',
  'naughty',
  'painter',
  'pretty',
  'sister',
  'small',
  'sweet'},
 'AF': {'confident', 'humour', 'kind', 'lame', 'lovable', 'short',
'sports'},
 'Sudeshna': {'bold',
  'childish',
  'chill',
  'confident',
  'eyes',
  'fair',
  'mature',
  'movies',
  'short'},
 'AH': {'funny', 'good character', 'kind', 'teddy bear'},
 'Nikhil': {'animal lover',
  'anime',
  'artistic',
  'drive',
  'fair',
  'introvert',
  'shy',
  'sleepy',
  'sports',
  'thin',
```

```
 'video game'},
'Priyanshu': {'badminton',
 'big face',
 'cricket',
 'driver',
 'fast thinking',
 'helping nature',
 'introvert',
 'lazy',
 'singing',
 'skin',
 'small',
 'sport',
 'weak communication'},
'AK': {'beutiful',
 'decision making',
 'emotional',
 'fearless',
 'good cook',
 'good dressing',
 'hairstylist',
 'humour',
 'intelligent',
 'introvert',
 'knowledge',
 'singer',
 'sketcher',
 'strong'},
'Radhika': {'beautiful',
 'best cook',
 'caring',
 'friend',
 'hardworking',
 'helping',
 'inspiring',
 'loving',
 'motivator',
 'patience',
 'sacrifice',
 'strength',
 'strong',
 'support'},
'Sachin Tendulkar': {'famous', 'fast', 'legendary', 'short'},
'Elon musk': {'business',
 'innovative',
 'multi billonaire',
 'space exploration'},
'Prashant': {'athletic',
 'bhajan',
```

```
  'flute',
  'honest',
  'lazy',
  'polite',
  'sleepy'},
 'Purna': {'newspaper', 'overthinker', 'puntual', 'teacher'},
 'Varun': {'determined',
  'disciplined',
  'glasses',
  'gym',
  'hard',
  'ignorant',
  'jolly',
  'kind',
  'orthodox',
  'physicque',
  'serious'},
 'Saket': {'attractive',
  'bold',
  'calm',
  'consistent',
  'destructive',
  'fit',
  'hard',
  'hypertrophy',
  'mental',
  'natural',
  'non controversial',
  'socially dead',
  'strong'},
 'Mr More': {'confident',
  'disciplined',
  'hardworking',
  'helping',
  'honest',
  'humble',
  'strict',
  'tall'},
 'Divyansh': {'conservative', 'disciplined', 'helpful', 'introvert'},
 'SVS Rohit': {'calm',
  'disciplined',
  'hardworking',
  'honest',
  'humble',
  'kind',
  'tall'},
 'Euchiro Oda': {'animated show',
  'manga comic',
  'netflix',
```

```
  'pirate',
  'story writer'},
 'Sudarshan': {'good character', 'workout'},
 'Sashank': {'cool',
  'enjoyable',
  'hardworking',
  'humour',
  'non vegetarian',
  'social'},
 'venkat': {'cool',
  'happy',
  'hardworking',
  'humour',
  'non vegetarian',
  'social'},
 'Theophila': {'coder', 'friendly', 'piano', 'reading', 'singer',
'writing'},
 'Saurav': {'bargaining',
  'best friend',
  'bulky body',
  'caring',
  'cricket',
  'expressive',
  'funny',
  'gentle',
  'gym',
  'table tennis',
  'traveling'},
 'AL': {'englandactor'},
 'Kesav1': {'badminton',
  'consistent',
  'cricket',
  'joyful',
  'short tempered',
  'space enthusiast',
  'tallest'},
 'Kesav2': {'friendly', 'vizag'}}

distances = compute_pairwise_distances(person_attribute_dict)

distances

[0.8888888888888888,
 0.8333333333333334,
 1.0,
 0.8571428571428572,
 0.9333333333333333,
 0.9130434782608696,
 0.8666666666666667,
 1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
0.9285714285714286,
1.0,
1.0,
1.0,
0.9444444444444444,
0.9047619047619048,
0.9615384615384616,
0.9411764705882353,
0.9473684210526316,
0.8461538461538461,
1.0,
1.0,
1.0,
0.9583333333333334,
1.0,
1.0,
1.0,
1.0,
0.9,
0.9565217391304348,
1.0,
1.0,
0.9411764705882353,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9,
1.0,
0.9411764705882353,
1.0,
0.8823529411764706,
1.0,
0.8421052631578947,
1.0,
1.0,
0.9285714285714286,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
0.9375,
0.95,
0.9583333333333334,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9545454545454546,
1.0,
1.0,
0.9333333333333333,
1.0,
0.9473684210526316,
0.9523809523809523,
0.9375,
1.0,
0.9333333333333333,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
0.9545454545454546,
0.9285714285714286,
0.9565217391304348,
0.8571428571428572,
1.0,
1.0,
0.8666666666666667,
1.0,
1.0,
1.0,
0.9230769230769231,
1.0,
1.0,
1.0,
0.9411764705882353,
0.8421052631578947,
```

```
0.9166666666666666,
0.9375,
0.9444444444444444,
1.0,
1.0,
1.0,
0.9090909090909091,
0.6666666666666667,
0.9230769230769231,
1.0,
1.0,
1.0,
0.95,
0.9047619047619048,
0.9411764705882353,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.95,
1.0,
0.9375,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9166666666666666,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9090909090909091,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9411764705882353,
0.92,
0.9444444444444444,
1.0,
1.0,
1.0,
0.9333333333333333,
1.0,
1.0,
0.9375,
1.0,
1.0,
1.0,
0.8947368421052632,
0.9130434782608696,
0.9259259259259259,
0.9473684210526316,
1.0,
0.8666666666666667,
1.0,
1.0,
0.9615384615384616,
0.9615384615384616,
1.0,
1.0,
0.9473684210526316,
1.0,
0.9565217391304348,
1.0,
0.95,
```

```
1.0,
0.8888888888888888,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9090909090909091,
1.0,
1.0,
1.0,
0.9444444444444444,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9375,
1.0,
0.9090909090909091,
0.9230769230769231,
0.875,
0.9333333333333333,
1.0,
1.0,
1.0,
0.875,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9333333333333333,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9166666666666666,
1.0,
1.0,
1.0,
0.9411764705882353,
1.0,
1.0,
1.0,
0.962962962962963,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9523809523809523,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9583333333333334,
1.0,
1.0,
1.0,
1.0,
0.9166666666666666,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
0.9230769230769231,
0.9411764705882353,
0.9,
1.0,
1.0,
1.0,
1.0,
0.9444444444444444,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9375,
1.0,
0.9166666666666666,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9230769230769231,
1.0,
0.9230769230769231,
0.9230769230769231,
0.9,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9230769230769231,
0.9230769230769231,
0.9,
1.0,
1.0,
1.0,
1.0,
0.9444444444444444,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.95,
0.95,
1.0,
1.0,
1.0,
1.0,
0.9411764705882353,
1.0,
0.9285714285714286,
0.9,
0.9230769230769231,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.8333333333333334,
0.9,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9333333333333333,
0.9473684210526316,
0.9,
1.0,
0.8571428571428572,
```

```
1.0,
1.0,
1.0,
0.9411764705882353,
1.0,
1.0,
1.0,
1.0,
0.9285714285714286,
1.0,
1.0,
1.0,
0.9,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9523809523809523,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.875,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9333333333333333,
1.0,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
0.9230769230769231,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9473684210526316,
0.8571428571428572,
1.0,
0.9375,
1.0,
1.0,
1.0,
0.9523809523809523,
0.9523809523809523,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9444444444444444,
1.0,
1.0,
1.0,
0.9230769230769231,
1.0,
1.0,
0.9333333333333333,
1.0,
1.0,
0.96,
0.8695652173913043,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
0.9444444444444444,
1.0,
1.0,
0.9411764705882353,
0.9411764705882353,
0.9411764705882353,
0.9047619047619048,
1.0,
1.0,
0.9230769230769231,
1.0,
1.0,
1.0,
1.0,
0.9642857142857143,
0.9655172413793104,
0.9285714285714286,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9523809523809523,
1.0,
0.9615384615384616,
1.0,
1.0,
1.0,
0.8571428571428572,
0.9,
0.9411764705882353,
1.0,
0.95,
1.0,
0.9,
1.0,
1.0,
```

```
1.0,
0.9411764705882353,
1.0,
0.9285714285714286,
1.0,
0.9230769230769231,
1.0,
1.0,
0.9166666666666666,
0.9166666666666666,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
0.9166666666666666,
1.0,
1.0,
1.0,
1.0,
0.9523809523809523,
0.9375,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9285714285714286,
1.0,
1.0,
```

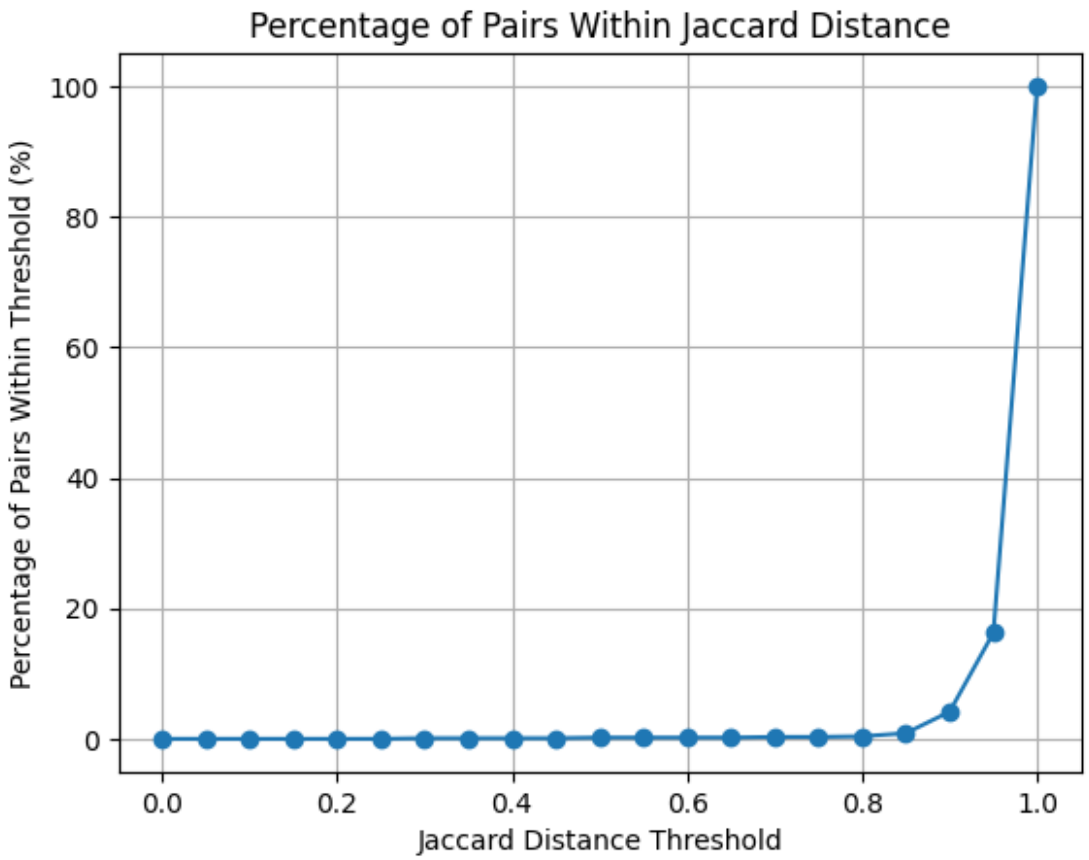
```
1.0,
0.9,
1.0,
0.8,
1.0,
1.0,
1.0,
0.9285714285714286,
1.0,
1.0,
1.0,
0.9565217391304348,
0.9583333333333334,
1.0,
1.0,
1.0,
0.9411764705882353,
1.0,
1.0,
1.0,
1.0,
0.9285714285714286,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9615384615384616,
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
0.9375,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9565217391304348,
```

```
1.0,
0.8888888888888888,
1.0,
0.962962962962963,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9615384615384616,
1.0,
0.9411764705882353,
1.0,
1.0,
1.0,
0.9473684210526316,
0.9473684210526316,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9615384615384616,
0.9,
1.0,
0.95,
1.0,
1.0,
0.9473684210526316,
0.9473684210526316,
1.0,
0.9583333333333334,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

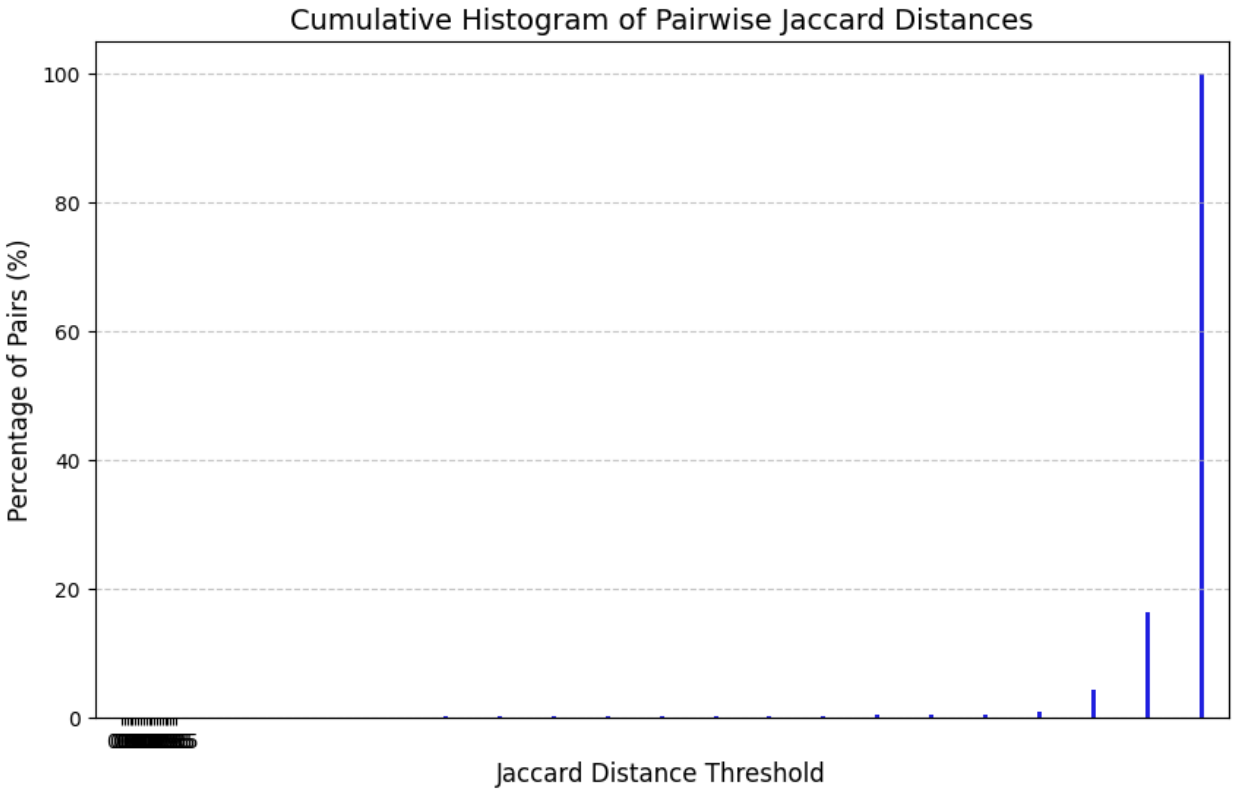
```
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9285714285714286,
1.0,
0.9230769230769231,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
1.0,
1.0,
1.0,
1.0,
0.9565217391304348,
0.9444444444444444,
0.9285714285714286,
0.875,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9523809523809523,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9473684210526316,
1.0,
0.9090909090909091,
0.5,
1.0,
1.0,
0.9230769230769231,
0.9230769230769231,
1.0,
1.0,
1.0,
1.0,
1.0,
0.9,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
```

```
 1.0,
 1.0,
 1.0,
 0.9166666666666666,
 0.9166666666666666,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 ...]

thresholds = np.arange(0, 1.05, 0.05)

percentages = compute_percentage_of_pairs(thresholds, distances)

x = thresholds
y = percentages
x_label = 'Jaccard Distance Threshold'
y_label = 'Percentage of Pairs Within Threshold (%)'
title = 'Percentage of Pairs Within Jaccard Distance'

plot_linear_plot(x, y, x_label, y_label, title)
```




```
plt.figure(figsize=(10, 6))
sns.barplot(x=thresholds, y=percentages, color="blue", width=0.08)

# Labels and Titles
plt.xlabel("Jaccard Distance Threshold", fontsize=12)
plt.ylabel("Percentage of Pairs (%)", fontsize=12)
plt.title("Cumulative Histogram of Pairwise Jaccard Distances",
fontsize=14)
plt.xticks(thresholds)  # Ensure all bins are labeled
plt.grid(axis="y", linestyle="--", alpha=0.7)

# Show plot
plt.show()
```


Cumulative Histogram of Pairwise Jaccard Distances
Percentage of Pairs (%)
100
80
60
40
20
0
Jaccard Distance Threshold