Instruction Manual

Experiment 5- Numerical Measure

Following Data given as a Longitudinal and Latitudinal Distance between Two Cities of globe. With the help of following table you have to compute distance between two cities using- Euclidean, Minkowski, Manhatten, Mahalonobis and Bhattacharya Distance.

Data :-

Longitude	Latitude	City Name
73	17	Mumbai
55	25	Dubai
0.12	51	London
13	53	Berlin
38	56	Moscow
32	116	Perth
140	36	Tokyo
74	40.7	New York

1. **Step1: Compute Distance based with Pen and Paper method.**
2. **Step2: Write a Python code for the same**
3. **Step3: Corroborate the Truth with analogy between Step 1 == Step 2**

Step1:

Euclidean Distance:-

$$d(i,j) = \sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\dots+(x_{ip}-x_{jp})^2}$$

∴ Distance between Mumbai and Dubai:-

$$=\sqrt{(73-55)^2+(17-25)^2}$$

$$= \sqrt{18^2+8^2} = \sqrt{324+64} = \sqrt{388} = \boxed{19.7}$$

Manhattan Distance:-

$$d(i,j) = |x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+\dots+|x_{ip}-x_{jp}|$$

∴ Manhattan Distance between Mumbai and Dubai:-

$$= |73-55|+|17-25|$$

$$= 18+8$$

$$= \boxed{26}$$

Minkowski Distance:-

Distance between Mumbai and Dubai:-

$$d(i,j) = \sqrt[p]{|x_{i1}-x_{j1}|^p+|x_{i2}-x_{j2}|^p+\dots+|x_{in}-x_{jn}|^p}$$

$= \sqrt[3]{|73-55|^3 + |17-25|^3}$ (where $p = 3$)

$= \sqrt[3]{18^3 + 8^3}$

$= \sqrt[3]{6344}$

$= \boxed{18.51}$ (where $p=3$)

Mahalanobis Distance : The Mahalanobis distance is a metric used to measure the distance between two points in a multidimensional space, taking into account the correlations among the variables. It is often used in statistics and data analysis to quantify the distance between data points while considering the covariance structure of the data.

The formula for Mahalanobis distance between two points, x and y, in a dataset with covariance matrix Σ, is given by:

$$D(x, y) = \sqrt{(x-y)^T \cdot \Sigma^{-1} \cdot (x-y)}$$

Example:

Suppose we have a dataset of three points with two features each: $X = \{(2,3), (4,6), (6,9)\}$. We want to calculate the Mahalanobis distance between the first point $(2,3)$ and the second point $(4,6)$.

Data Preparation:

The dataset can be represented as follows:

$$X = \begin{bmatrix} 2 & 3 \\ 4 & 6 \\ 6 & 9 \end{bmatrix}$$

2. **Covariance Matrix Calculation**:

We need to compute the covariance matrix of the dataset. The covariance matrix S is calculated as follows:

$$S = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$$

where n is the number of data points and $\bar{x}$ is the mean vector of the data.

In our case, $n = 3$ and $\bar{x} = (4, 6)$. Therefore,

$$S = \frac{1}{2}\begin{bmatrix} 2 & 3 \\ 2 & 3 \end{bmatrix}$$

---

3. **Mahalanobis Distance Calculation**:

The Mahalanobis distance between two points x and y using the covariance matrix S is given by:

$$D(x, y) = \sqrt{(x - y)^T \cdot S^{-1} \cdot (x - y)}$$

For our example, $x = (2, 3)$ and $y = (4, 6)$. Plugging in the values, we have:

$$D((2,3),(4,6)) = \sqrt{\begin{bmatrix} -2 & -3 \end{bmatrix} \cdot \begin{bmatrix} 1 & -1 \\ -1 & 1.5 \end{bmatrix} \cdot \begin{bmatrix} -2 \\ -3 \end{bmatrix}}$$

Calculating the matrix multiplications gives:

$$D((2,3),(4,6)) = \sqrt{\begin{bmatrix} 1.5 & -1 \end{bmatrix} \cdot \begin{bmatrix} -2 \\ -3 \end{bmatrix}} = \sqrt{9+3} = \sqrt{12} \approx 3.464$$

So, the Mahalanobis distance between the points (2,3)(2,3) and (4,6)(4,6) in this example is approximately 3.4643.464.

The Bhattacharyya distance is a measure of the similarity between two probability distributions. It's often used in pattern recognition, image processing, and statistics to quantify the difference between two distributions. The Bhattacharyya distance is calculated using the Bhattacharyya coefficient, which measures the overlap between the distributions. The formula for Bhattacharyya distance between two distributions P and Q is as follows:

$$D_B(P,Q) = -\ln(BC(P,Q))$$

Where $BC(P,Q)$ is the Bhattacharyya coefficient:

$$BC(P,Q) = \sum_i \sqrt{P(i) \cdot Q(i)}$$

Here, $P(i)$ and $Q(i)$ are the probabilities of occurrence of the i-th event in distributions P and Q, respectively. The sum is taken over all possible events.

Example: There are 2 event Pa and Q . Its probanility given are as follows:

Distribution P:| Event | Probability

| A | 0.3 |

| B | 0.4 |

| C | 0.2 |

| D | 0.1 |

Distribution Q:| Event | Probability |

| A | 0.2 |

| B | 0.3 |

| C | 0.3 |

| D | 0.2 |

Then compute Bhattacharya Distance ?

1. Calculate the Bhattacharyya coefficient $BC(P,Q)$:

$$BC(P,Q) = \sqrt{0.3 \cdot 0.2} + \sqrt{0.4 \cdot 0.3} + \sqrt{0.2 \cdot 0.3} + \sqrt{0.1 \cdot 0.2}$$
$$BC(P,Q) \approx 0.8416$$

1. Compute the Bhattacharyya distance $D_B(P,Q)$:

$$D_B(P,Q) = -\ln(BC(P,Q)) \approx 0.176$$

So, the Bhattacharyya distance between distributions P and Q is approximately 0.176.

Step2: Code:-

1. Importing Libraries and creating dataset:-

```
[1] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
```

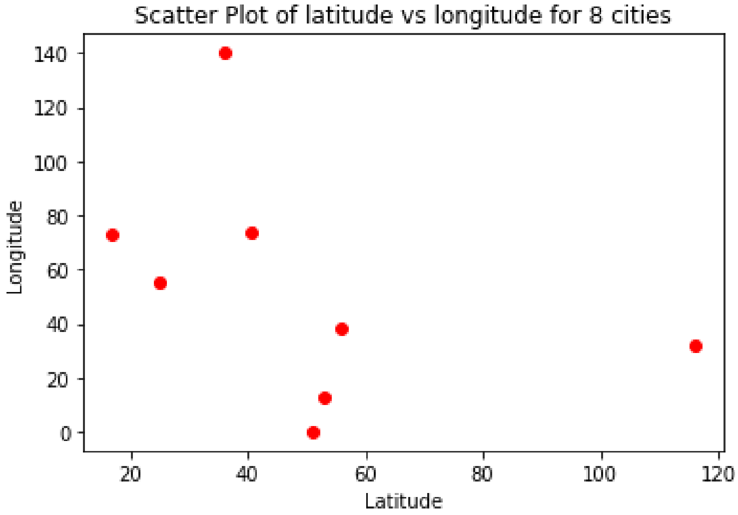
```
data = {
    'longitude': [73, 55, 0.12, 13, 38, 32, 140, 74],
    'latitude': [17, 25, 51, 53, 56, 116, 36, 40.7],
    'city': ['Mumbai', 'Dubai', 'London', 'Berlin', 'Moscow', 'Perth', 'Tokyo', 'New York']
}

df = pd.DataFrame(data)
df
```

	longitude	latitude	city
0	73.00	17.0	Mumbai
1	55.00	25.0	Dubai
2	0.12	51.0	London
3	13.00	53.0	Berlin
4	38.00	56.0	Moscow
5	32.00	116.0	Perth
6	140.00	36.0	Tokyo
7	74.00	40.7	New York

2. Scatter Plot:-

```python
[3] plt.scatter(df["latitude"], df["longitude"], c ="red")
    plt.xlabel("Latitude")
    plt.ylabel("Longitude")
    plt.title("Scatter Plot of latitude vs longitude for 8 cities")
    plt.show()
```



2. Euclidean Distance:-

```python
[4] def euclidean(info):
      dist = []
      for _,i in df.iterrows():
        d = ((i.latitude - info.latitude) ** 2 + (i.longitude - info.longitude) **2 ) ** 0.5
        dist.append(round(d, 2))
      return dist
```

```python
[5] df_euclidean = df.copy(deep=True)
    for _,i in df.iterrows():
      df_euclidean[f"Euclidian Distance from {i.city}"] = euclidean(i)
```

```
[6] df_euclidean
```

	longitude	latitude	city	Euclidian Distance from Mumbai	Euclidian Distance from Dubai	Euclidian Distance from London	Euclidian Distance from Berlin	Euclidian Distance from Moscow	Euclidian Distance from Perth	Euclidian Distance from Tokyo	Euclidian Distance from New York
0	73.00	17.0	Mumbai	0.00	19.70	80.42	69.97	52.40	107.15	69.64	23.72
1	55.00	25.0	Dubai	19.70	0.00	60.73	50.48	35.36	93.86	85.71	24.65
2	0.12	51.0	London	80.42	60.73	0.00	13.03	38.21	72.40	140.68	74.59
3	13.00	53.0	Berlin	69.97	50.48	13.03	0.00	25.18	65.80	128.13	62.23
4	38.00	56.0	Moscow	52.40	35.36	38.21	25.18	0.00	60.30	103.94	39.12
5	32.00	116.0	Perth	107.15	93.86	72.40	65.80	60.30	0.00	134.40	86.22
6	140.00	36.0	Tokyo	69.64	85.71	140.68	128.13	103.94	134.40	0.00	66.17
7	74.00	40.7	New York	23.72	24.65	74.59	62.23	39.12	86.22	66.17	0.00

3. Manhattan Distance:-

```
[10] def manhattan(info):
       dist = []
       for _,i in df.iterrows():
         d = (abs(i.latitude - info.latitude) + abs(i.longitude - info.longitude))
         dist.append(round(d, 2))
       return dist
```

```
[11] df_manhattan = df.copy(deep=True)
     for _,i in df.iterrows():
       df_manhattan[f"Manhattan Distance from {i.city}"] = manhattan(i)
```

```
[12] df_manhattan
```

	longitude	latitude	city	Manhattan Distance from Mumbai	Manhattan Distance from Dubai	Manhattan Distance from London	Manhattan Distance from Berlin	Manhattan Distance from Moscow	Manhattan Distance from Perth	Manhattan Distance from Tokyo	Manhattan Distance from New York
0	73.00	17.0	Mumbai	0.00	26.00	106.88	96.00	74.00	140.00	86.00	24.70
1	55.00	25.0	Dubai	26.00	0.00	80.88	70.00	48.00	114.00	96.00	34.70
2	0.12	51.0	London	106.88	80.88	0.00	14.88	42.88	96.88	154.88	84.18
3	13.00	53.0	Berlin	96.00	70.00	14.88	0.00	28.00	82.00	144.00	73.30
4	38.00	56.0	Moscow	74.00	48.00	42.88	28.00	0.00	66.00	122.00	51.30
5	32.00	116.0	Perth	140.00	114.00	96.88	82.00	66.00	0.00	188.00	117.30
6	140.00	36.0	Tokyo	86.00	96.00	154.88	144.00	122.00	188.00	0.00	70.70
7	74.00	40.7	New York	24.70	34.70	84.18	73.30	51.30	117.30	70.70	0.00

4. **Minkowski Distance:-**

```
[13] def minkowski(info):
       p = 3
       dist = []
       for _,i in df.iterrows():
         d = ((abs(i.latitude - info.latitude)) ** p + (abs(i.longitude - info.longitude)) ** p) ** (1/3)
         dist.append(round(d, 2))
       return dist
```

```
[14] df_minkowski = df.copy(deep=True)
     for _,i in df.iterrows():
       df_minkowski[f"Minkowski Distance from {i.city}"] = minkowski(i)
```

```
[15] df_minkowski
```

	longitude	latitude	city	Minkowski Distance from Mumbai	Minkowski Distance from Dubai	Minkowski Distance from London
0	73.00	17.0	Mumbai	0.00	18.51	75.27
1	55.00	25.0	Dubai	18.51	0.00	56.76
2	0.12	51.0	London	75.27	56.76	0.00
3	13.00	53.0	Berlin	64.04	45.79	12.90
4	38.00	56.0	Moscow	46.75	32.62	37.91
5	32.00	116.0	Perth	101.29	91.49	67.46
6	140.00	36.0	Tokyo	67.51	85.06	139.94
7	74.00	40.7	New York	23.70	22.06	73.95

Minkowski Distance from Berlin	Minkowski Distance from Moscow	Minkowski Distance from Perth	Minkowski Distance from Tokyo	Minkowski Distance from New York
64.04	46.75	101.29	67.51	23.70
45.79	32.62	91.49	85.06	22.06
12.90	37.91	67.46	139.94	73.95
0.00	25.01	63.57	127.10	61.17
25.01	0.00	60.02	102.26	36.90
63.57	60.02	0.00	121.00	79.43
127.10	102.26	121.00	0.00	66.01
61.17	36.90	79.43	66.01	0.00

MAHALONOBIS Distance :

```
import numpy as np

def mahalanobis_distance(x, mean, covariance):

    d = x - mean

    inv_covariance = np.linalg.inv(covariance)

    distance = np.sqrt(np.dot(np.dot(d.T, inv_covariance), d))

    return distance

# Example data

point = np.array([1.5, 2.0])

mean = np.array([1.0, 1.5])

covariance = np.array([[1.0, 0.5], [0.5, 1.0]])

distance = mahalanobis_distance(point, mean, covariance)

print("Mahalanobis Distance:", distance)
```

```
import numpy as np
import math
```

```
# Probability distributions P and Q
P = np.array([0.3, 0.4, 0.2, 0.1])
Q = np.array([0.2, 0.3, 0.3, 0.2])

def bhattacharyya_distance(p, q):
    # Calculate Bhattacharyya coefficient
    bc = np.sum(np.sqrt(p * q))

    # Calculate Bhattacharyya distance
    b_distance = -math.log(bc)
    return b_distance

# Compute the Bhattacharyya distance between P and Q
b_distance = bhattacharyya_distance(P, Q)
print("Bhattacharyya Distance:", b_distance)
```