pd5zs3uaa

April 29, 2024

[ ]:
```
import matplotlib.pyplot as plt
import numpy as np
import random
import math
```

[ ]:
```

```

Euclidean

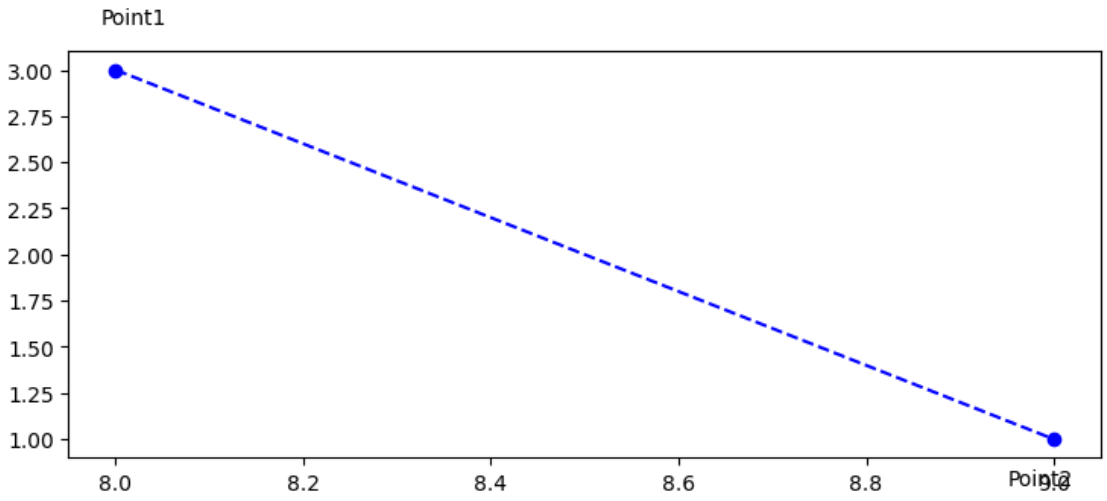
[ ]:
```
import numpy as np

# initializing points in
# numpy arrays
point1 = np.array((1, 2, 3))
point2 = np.array((1, 1, 1))

# calculating Euclidean distance
# using linalg.norm()
dist = np.linalg.norm(point1 - point2)
# printing Euclidean distance
print(dist)
```

```
2.23606797749979
```

Euclidean

[ ]:
```
plt.rcParams["figure.figsize"] = [7.50, 3.50]
plt.rcParams["figure.autolayout"] = True
point1 = [random.randint(0, 10), random.randint(0, 10)]
point2 = [random.randint(0, 10),random.randint(0, 10)]
x_values = [point1[0], point2[0]]
y_values = [point1[1], point2[1]]
e= pow(((pow((point2[0]-point1[0]),2))+(pow((point2[1]-point1[1]),2))),(1/2))
print("Euclidean Dintance from Point1 to Point2 = ", e)
plt.plot(x_values, y_values, 'bo', linestyle="--")
plt.text(point1[0]-0.015, point1[1]+0.25, "Point1")
plt.text(point2[0]-0.050, point2[1]-0.25, "Point2")
plt.show()
```

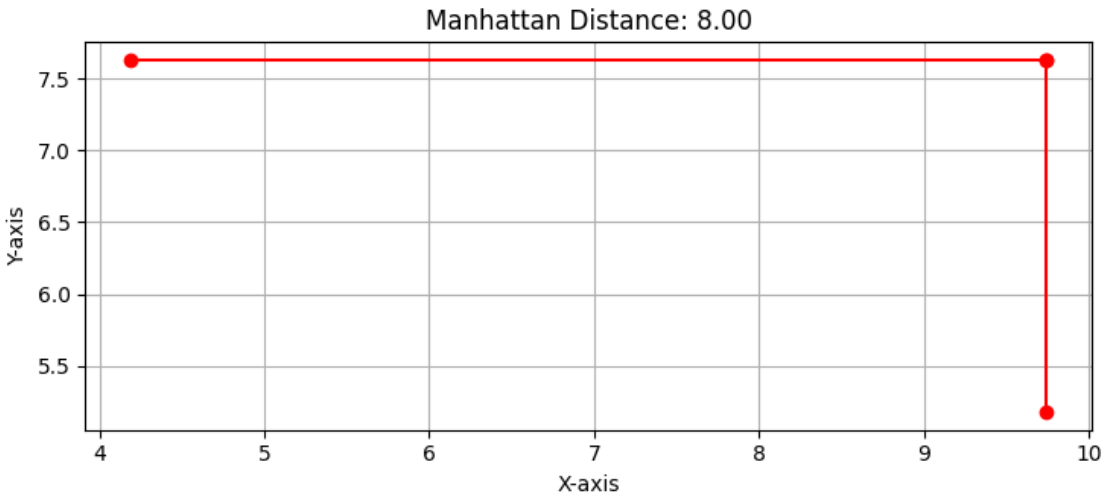
```
Euclidean Dintance from Point1 to Point2 =  2.23606797749979
```



[ ]:

Manhattan Distance

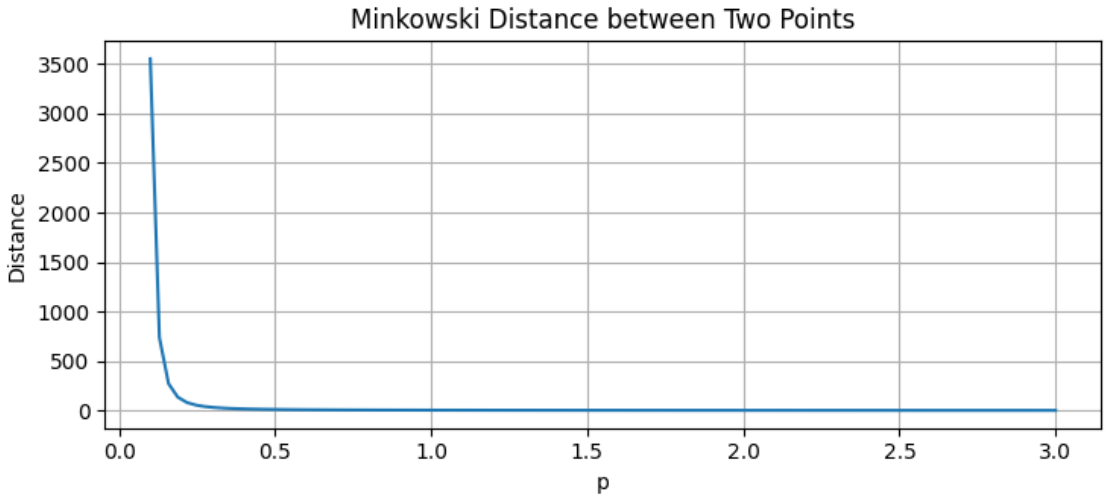
[ ]:
```
point1 = np.random.rand(2) * 10
point2 = np.random.rand(2) * 10
m = np.sum(np.abs(point1 - point2))
plt.plot([point1[0], point2[0]], [point1[1], point1[1]], 'ro-')
plt.plot([point2[0], point2[0]], [point1[1], point2[1]], 'ro-')
# plt.text(point1[0], point1[1], '  Point 1', verticalalignment='bottom',␣
 ↪horizontalalignment='right')
# plt.text(point2[0], point2[1], '  Point 2', verticalalignment='bottom',␣
 ↪horizontalalignment='right')
plt.title(f"Manhattan Distance: {m:.2f}")
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.grid(True)
plt.show()
```



Minkowski Distance

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt

def minkowski_distance(x1, y1, x2, y2, p):
    return ((abs(x2 - x1)**p + abs(y2 - y1)**p)**(1/p))
x1, y1 = 1, 1
x2, y2 = 4, 5
p_values = np.linspace(0.1, 3, 100)
distances = [minkowski_distance(x1, y1, x2, y2, p) for p in p_values]
print("minkowski distance = ",minkowski_distance(x1, y1, x2, y2, p))
plt.plot(p_values, distances)
plt.title('Minkowski Distance between Two Points')
plt.xlabel('p')
plt.ylabel('Distance')
plt.grid(True)
plt.show()
```

```
4.497941445275415
```



Minkowski Distance

[ ]:
```
from math import *
from decimal import Decimal
def p_root(value, root):
    root_value = 1 / float(root)
    return round (Decimal(value) **
             Decimal(root_value), 3)

def minkowski_distance(x, y, p_value):
    return (p_root(sum(pow(abs(a-b), p_value)
            for a, b in zip(x, y)), p_value))

vector1 =  np.random.randint(1,101,5)
vector2 =  np.random.randint(1,101,5)
p = 3
print("Minkowski Distance = ",minkowski_distance(vector1, vector2, p))
```

```
Minkowski Distance =  67.056
```

Cosine Similarity

[ ]:
```
from sklearn.metrics.pairwise import cosine_similarity

A = np.array([np.random.randint(1,101,5)])
B = np.array([np.random.randint(1,101,5)])

cosine_similarity_result = cosine_similarity(A, B)
print(f"Cosine Similarity = {cosine_similarity_result[0][0]}")
```

```
Cosine Similarity = 0.9496009006146451
```