ONLINE DETECTION

DETECTION MODEL

ESTUDILLO | TABASA

The demonstration website showcases our KNN model in a real-world scenario, featuring:

A 25-question multiple-choice quiz interface

Real-time monitoring of behavioral features:

- Browser tab switching
- Time spent per question
- Keyboard Activity
- Mouse Clicks
- Face Proximity
- Inactivity Periods
- Window Switching Pattern

USUAL STEPS IN KNN ALGORITHMS

Step 1: Choose the number ***k***, representing how many nearest neighbors to consider.

Step 2: Compute the distance of the unlabeled point ***z*** to all points in the training dataset ***D***.

Step 3: Sort the distances from ***z*** to all training points.

Step 4: Select the ***k*** closest neighbors.

Step 5: Assign ***z*** to the majority class of the kkk nearest neighbors.

KEY STEPS

1. PREPROCESSING
2. KNN IMPLEMENTATION
3. PERFORMANCE EVALUATION
4. VISUALIZATION
5. WEB IMPLEMENTATION

Learning Objectives

The goal is to classify students as "Cheating Detected" (**Class 1**) or "No Cheating Detected" (**Class 0**).

PREPROCESSING

We manually normalize the features to ensure consistent scaling, then split the dataset into 80% training and 20% testing. The dataset is artificially created to realistically simulate a 25-item test scenario.

FEATURES

- Browser tab switching
- Time spent per question
- Keyboard Activity
- Mouse Clicks
- Face Proximity
- Inactivity Periods
- Window Switching Pattern

KNN IMPLEMENTATION

We implemented a function to compute distances between points, as required by any standard KNN algorithm, starting with:

MANHATTAN DISTANCE

$$distance = \sum_{1}^{n} |p_i - q_i|$$

**calculates the total absolute differences, making it sensitive to variations in features with large ranges (e.g., mouse movements or clicks).

EUCLIDEAN DISTANCE

$$d(x,\ y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$

**emphasizes larger differences more heavily, being influenced by squared differences between features.

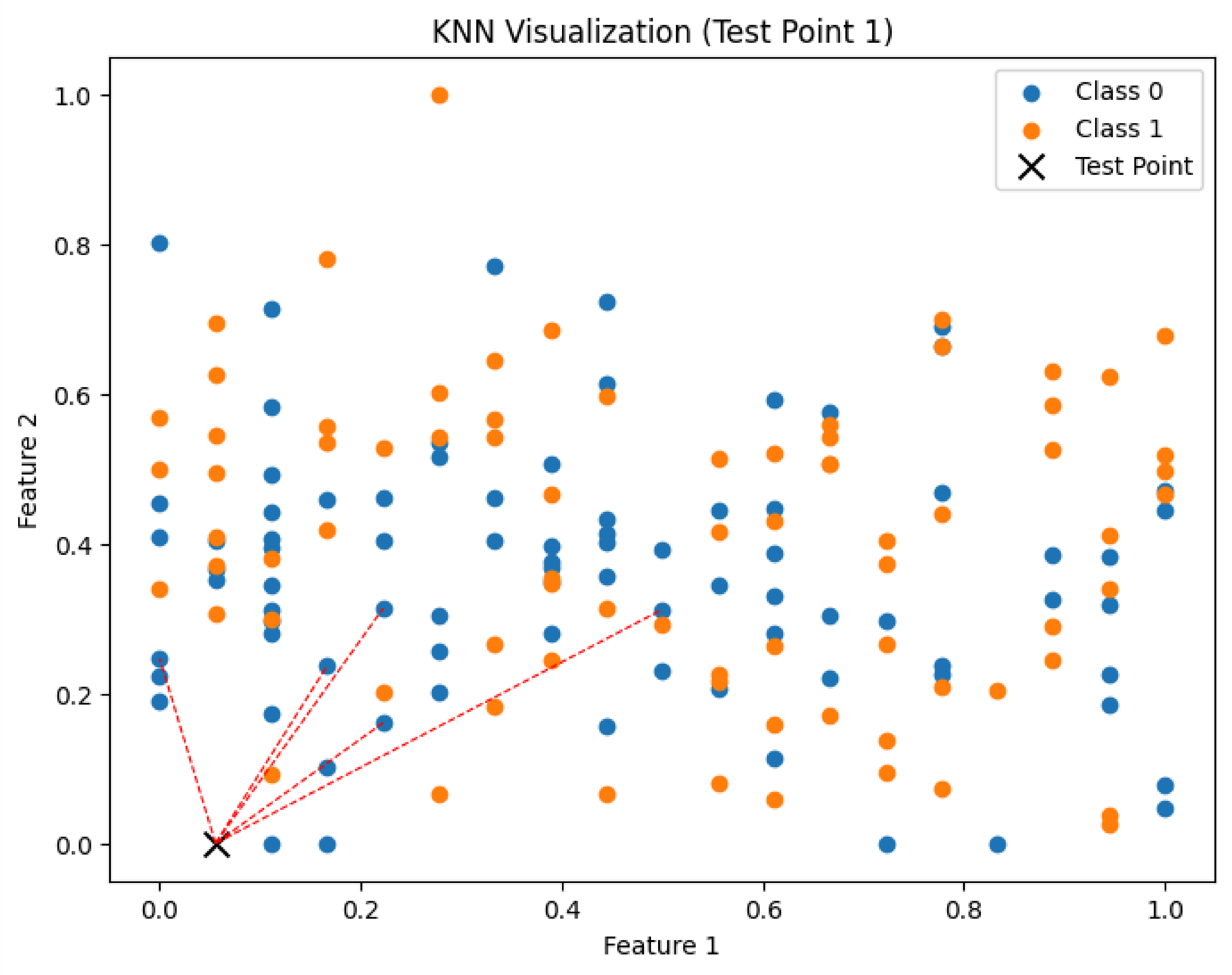
SUPREMUM DISTANCE

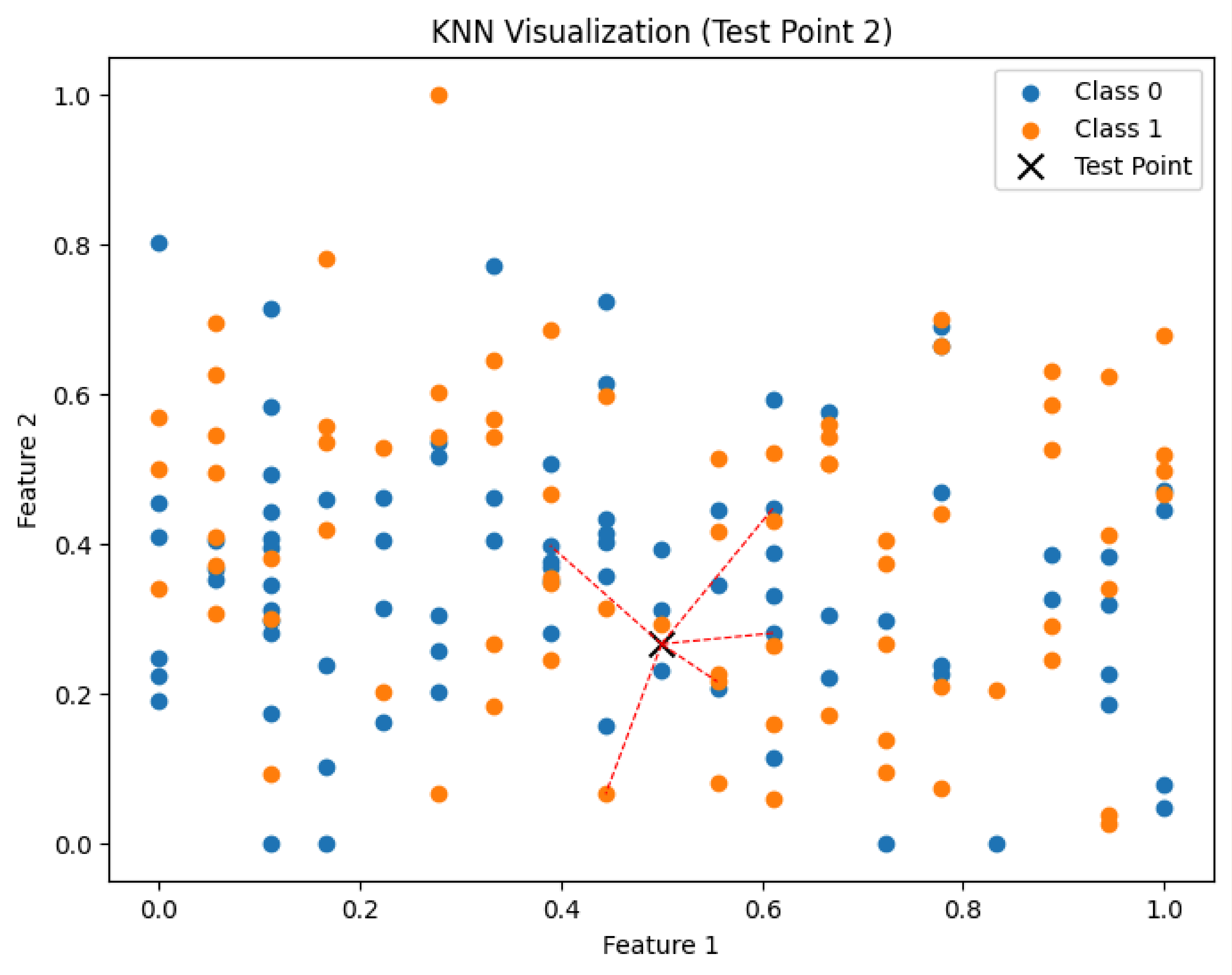
$$d(y,z) = \lim_{h\to\infty}\left(\sum_{f=1}^{p}\left|x_{yf} - x_{zf}\right|^{h}\right)^{\frac{1}{h}} = \max_{f}^{p} \left|x_{yf} - x_{zf}\right|$$

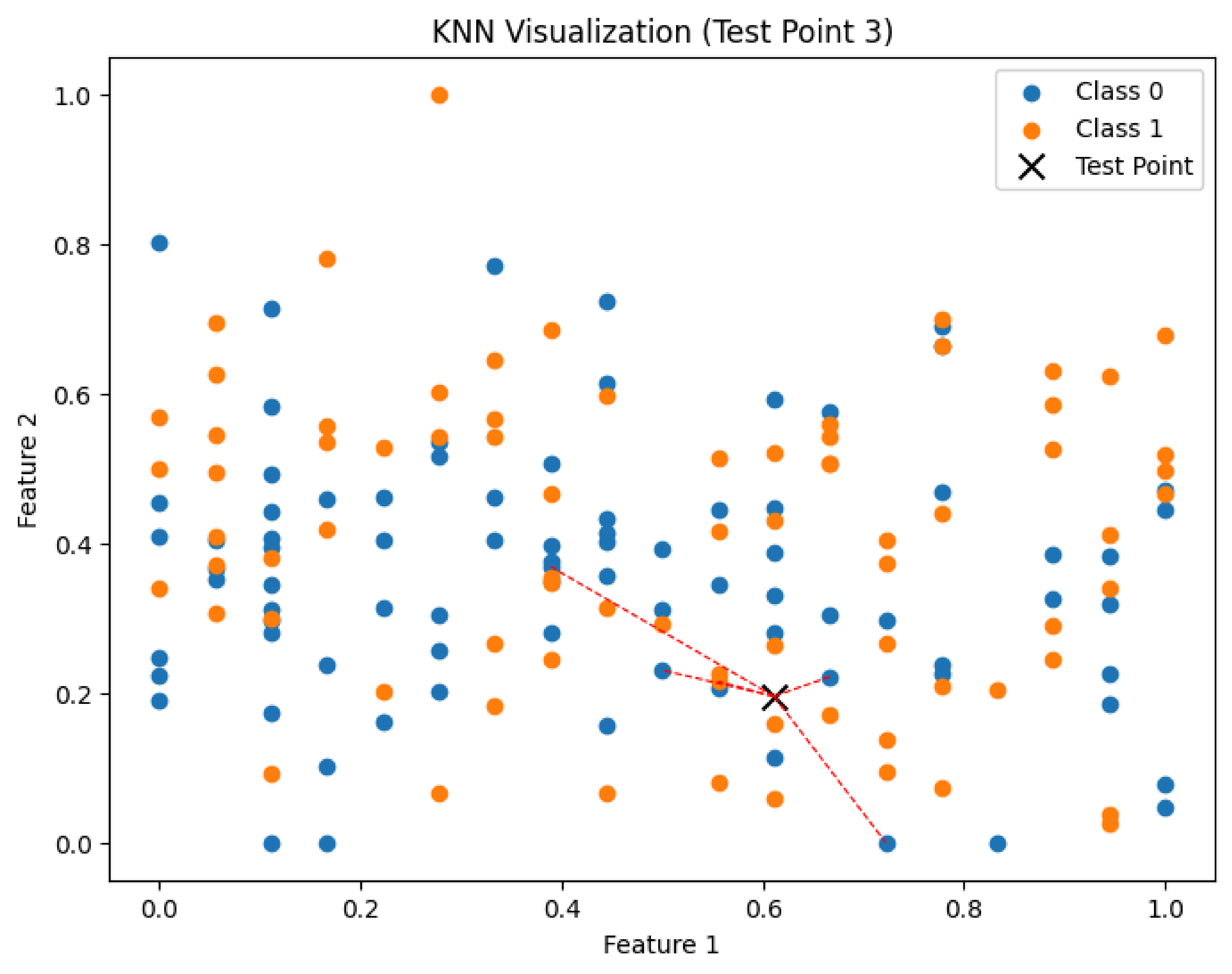
**only considers the largest absolute difference between features, ignoring cumulative variations.

VISUALIZATION OF KNN

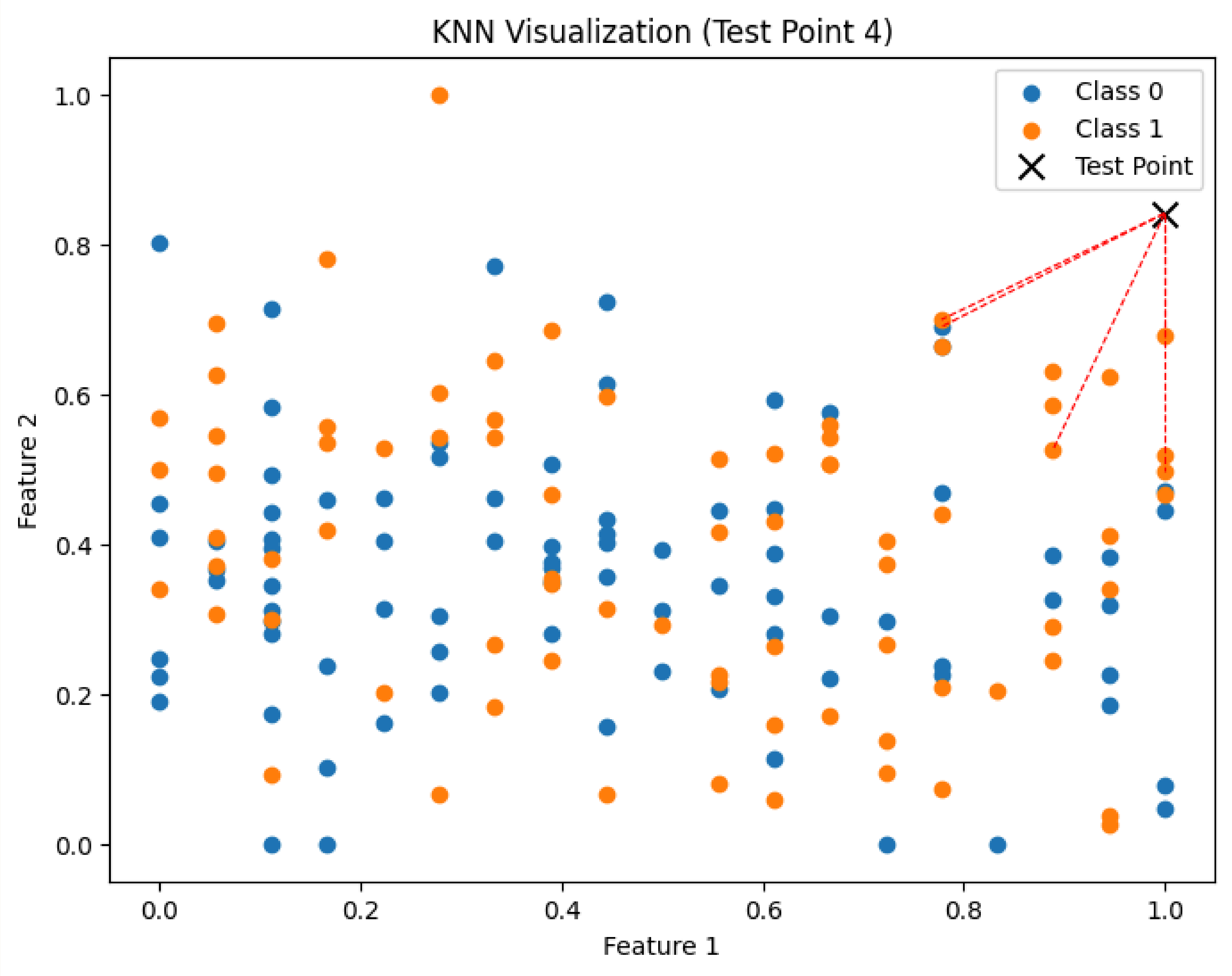
- **Training Points:** Plotted with unique colors for each class.
- **Test Point:** Highlighted (e.g., black "X").
- **Nearest Neighbors:** Connected to the test point using a distance metric (e.g., Euclidean).
- **Majority Vote:** Class assigned based on the highest neighbor vote.

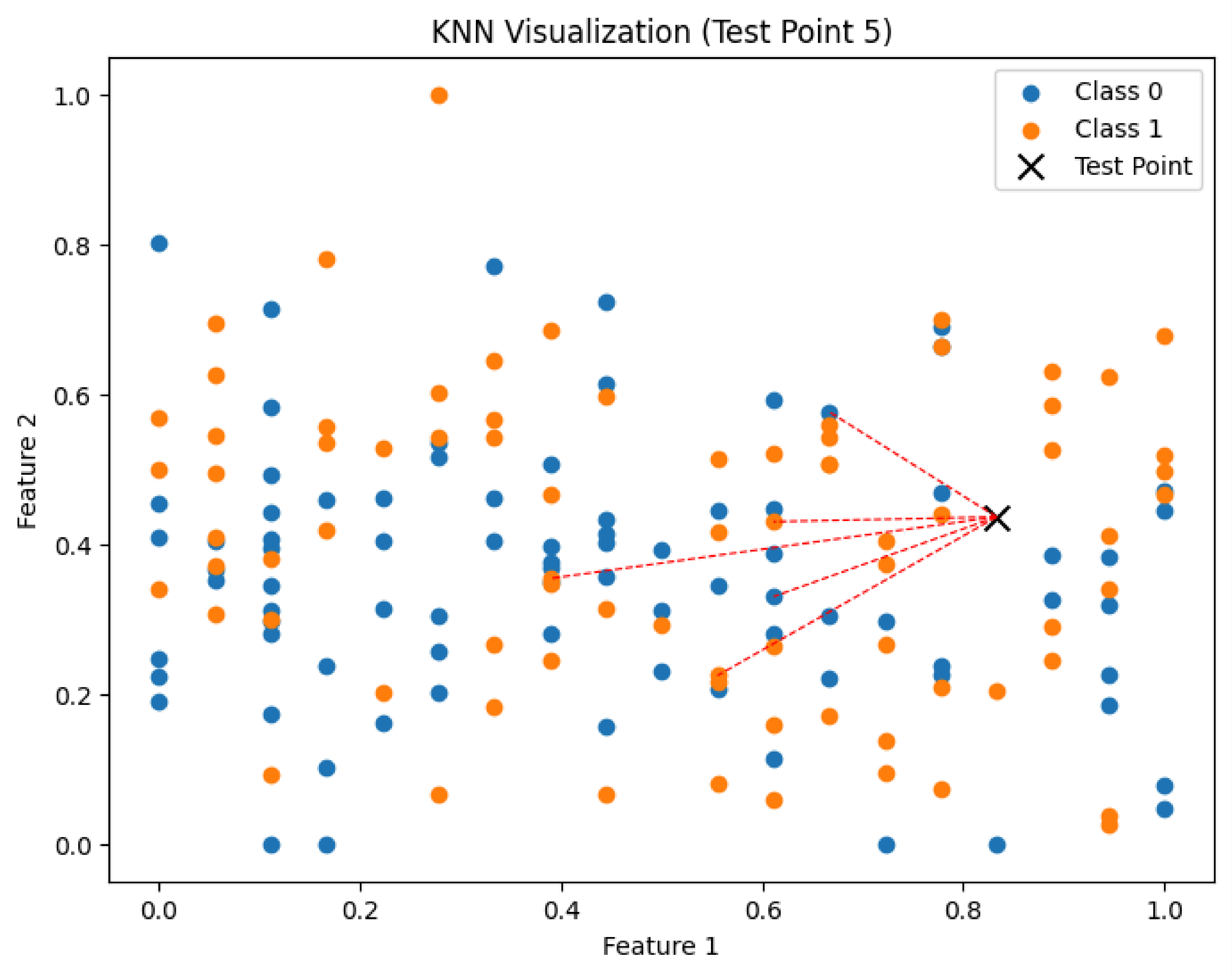

KNN Visualization (Test Point 1)
Class 0
Class 1
Test Point
Feature 1
Feature 2
0.0
0.2
0.4
0.6
0.8
1.0


KNN Visualization (Test Point 2)
Class 0
Class 1
Test Point
Feature 1
Feature 2
0.0
0.2
0.4
0.6
0.8
1.0



KNN Visualization (Test Point 3)
Class 0
Class 1
Test Point
Feature 1
Feature 2
0.0
0.2
0.4
0.6
0.8
1.0


KNN Visualization (Test Point 4)
Class 0
Class 1
Test Point
Feature 1
Feature 2
0.0
0.2
0.4
0.6
0.8
1.0



KNN Visualization (Test Point 5)
Class 0
Class 1
Test Point
Feature 2
Feature 1
0.0
0.2
0.4
0.6
0.8
1.0

KNN ACCURACY

67.50%

MANHATTAN

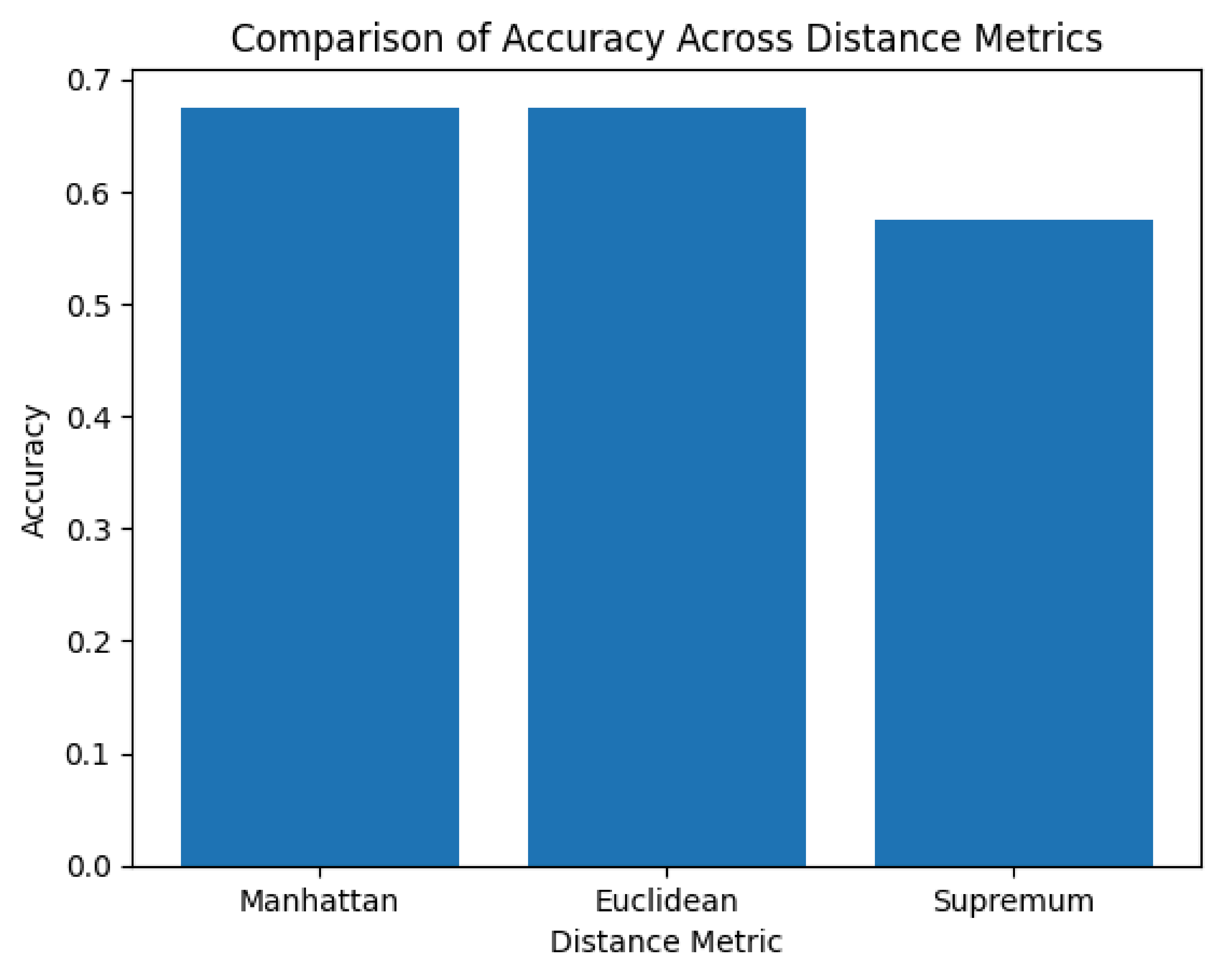
Accuracy: 67.50%
Precision: 0.80
Recall: 0.55
F1-Score: 0.65

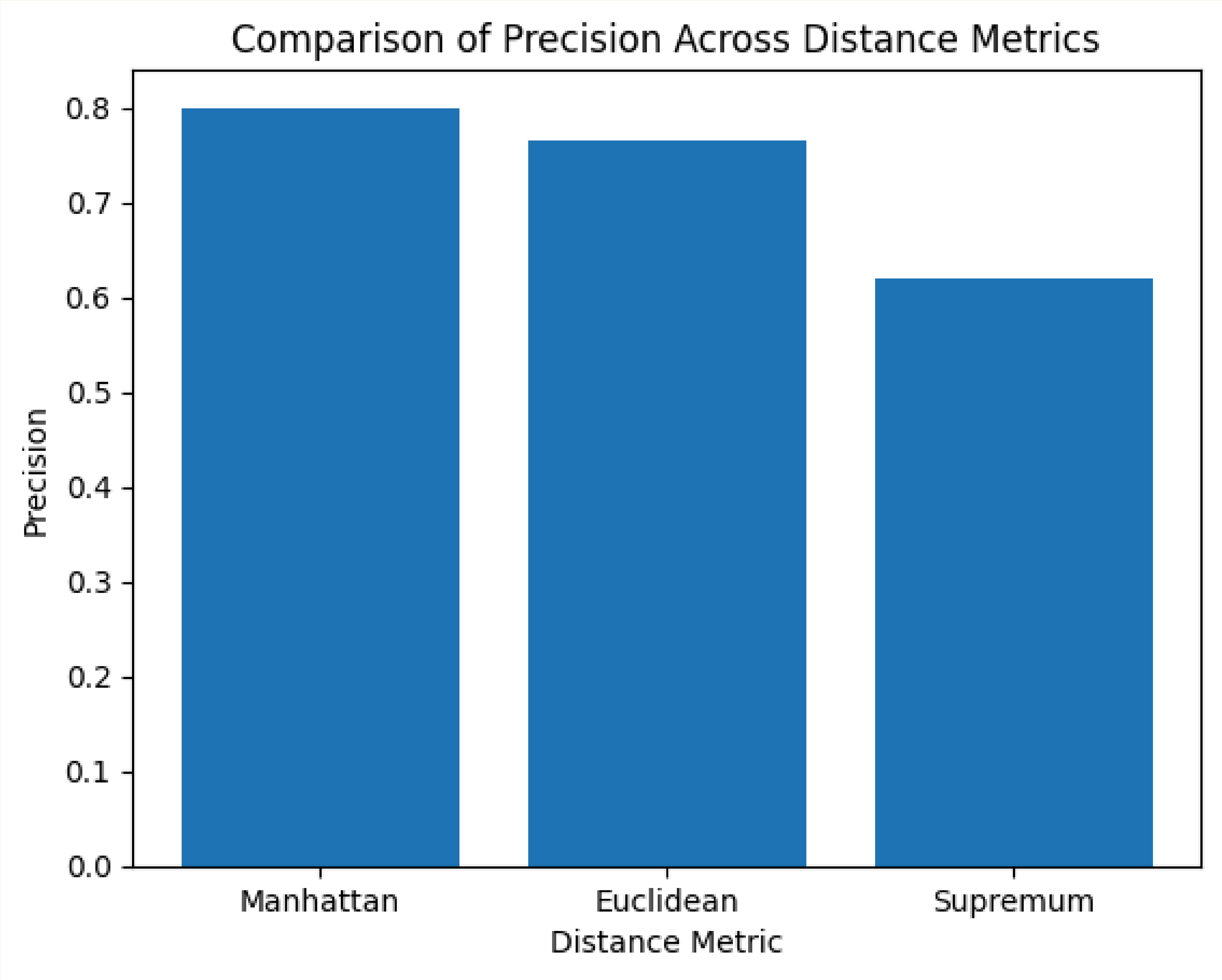
EUCLIDEAN

Accuracy: 67.50%
Precision: 0.76
Recall: 0.59
F1-Score: 0.67

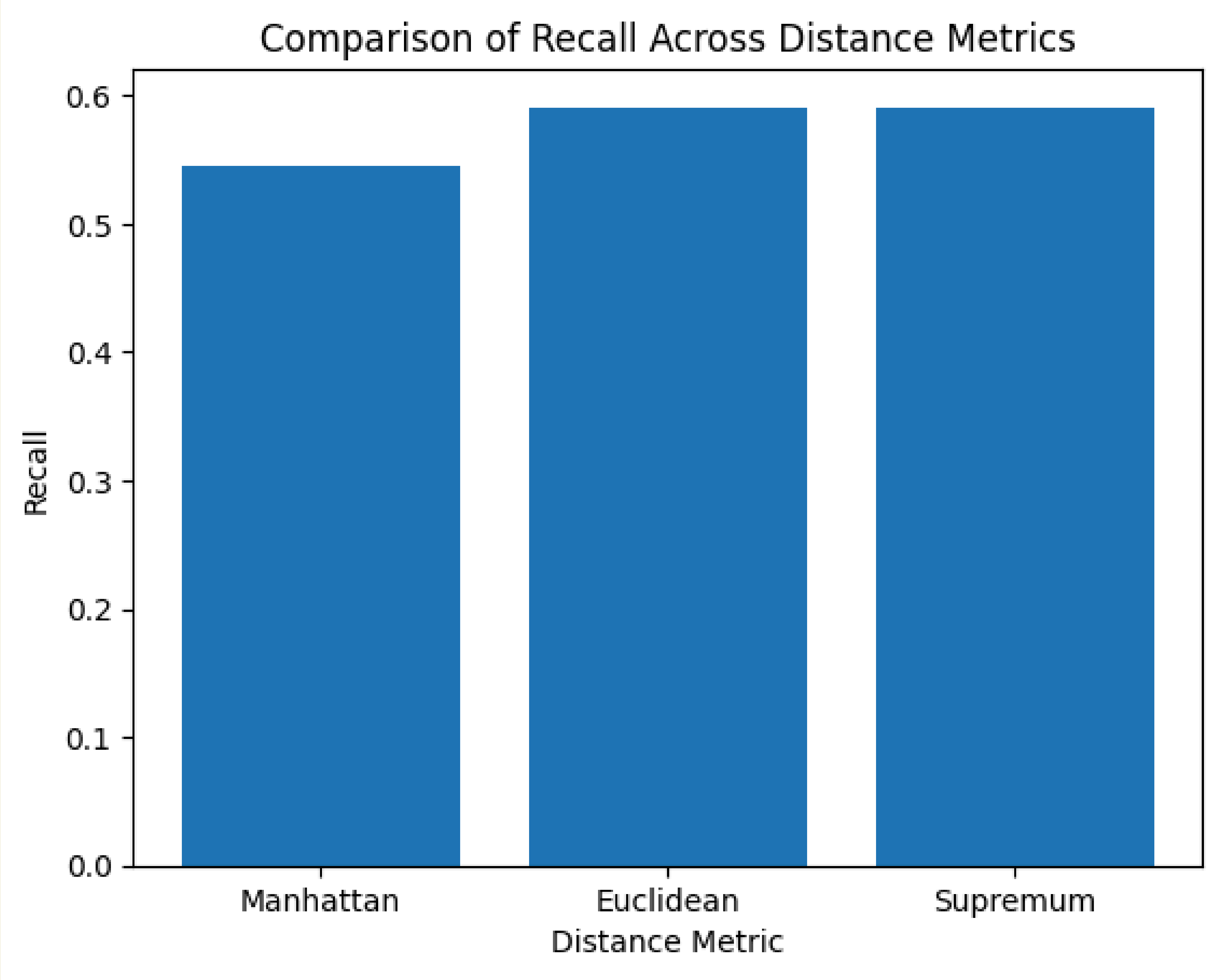
SUPREMUM

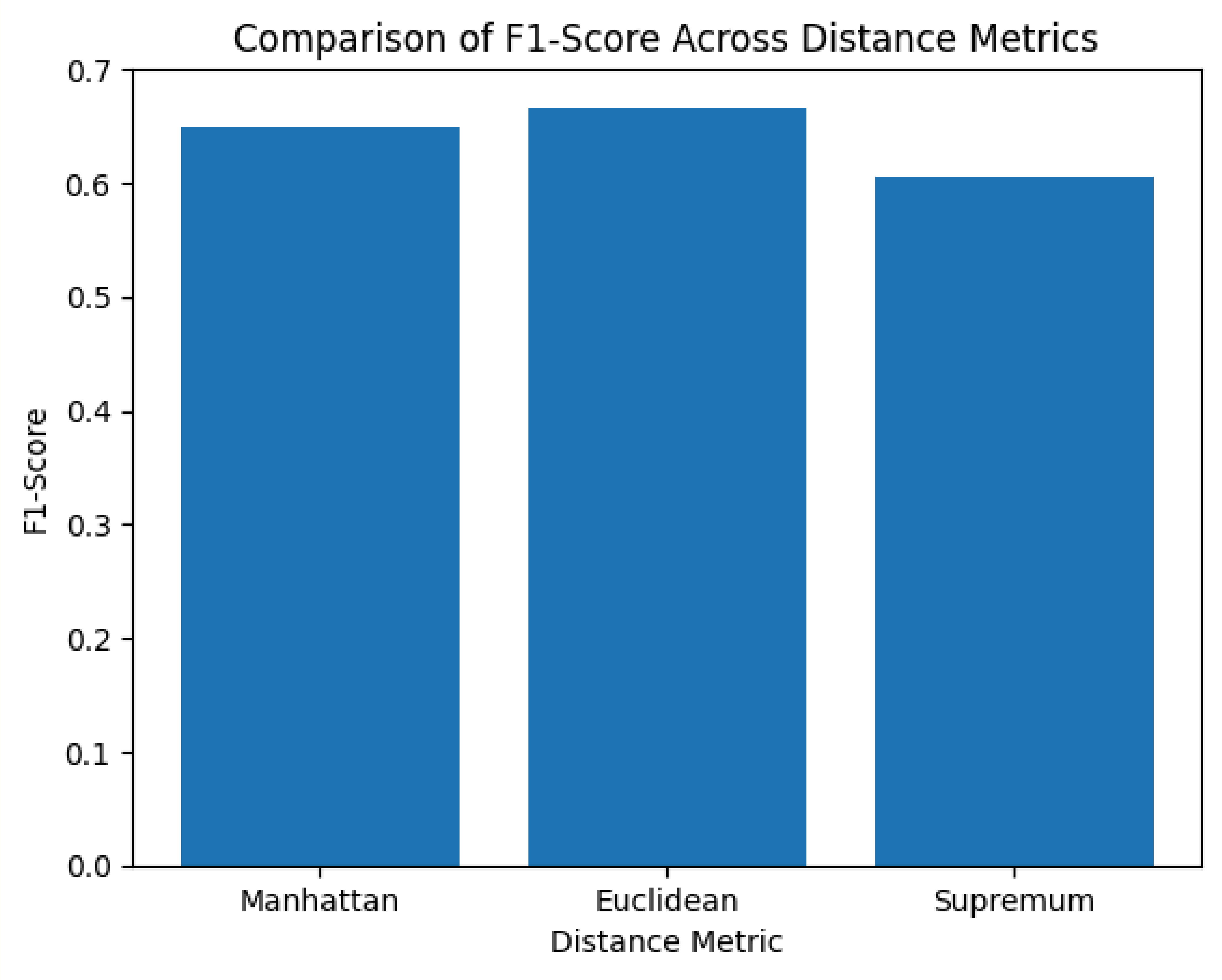
Accuracy: 67.50%
Precision: 0.62
Recall: 0.59
F1-Score: 0.60


Comparison of Accuracy Across Distance Metrics
Accuracy
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Manhattan
Euclidean
Supremum
Distance Metric



Comparison of Precision Across Distance Metrics
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Precision
Manhattan
Euclidean
Supremum
Distance Metric


Comparison of Recall Across Distance Metrics
Recall
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Manhattan
Euclidean
Supremum
Distance Metric


Comparison of F1-Score Across Distance Metrics
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
F1-Score
Manhattan
Euclidean
Supremum
Distance Metric

CONCLUSIONS

- **Manhattan Distance** is better for precision-focused tasks where false positives must be minimized.
- **Euclidean Distance** provides better balance between precision and recall, making it more suitable for general applications.
- **Supremum Distance** is not effective for this dataset due to its inability to account for cumulative feature variations.

CONCLUSIONS

1. **Feature Scales:**
   - Features like Keyboard_Activity and Mouse_Movements_or_Clicks have large ranges, dominating the distance calculation. Metrics sensitive to these scales perform better.
2. **Class Imbalance:**
   - Slight imbalance in Cheating_Detected cases affects recall, as fewer positive cases make it harder to capture all cheating instances.
3. **Feature Importance:**
   - Features such as Mouse_Movements_or_Clicks and Keyboard_Activity appear highly correlated with cheating. Distance metrics that account for these perform better.