Name: Abhay Sharma

Class: BE - C

Moodle Id: 20102065

Roll No.: 162

Sub.: Applied Data Science

Experiment No.: 08

Aim: Implement distance-based anomaly detection using k nearest neighbours.

Importing Libraries

```
from sklearn.neighbors import NearestNeighbors
import numpy as np
import matplotlib.pyplot as plt
```

Generating Sample Data

```
np.random.seed(42)
X = 0.3 * np.random.randn(100, 3)
X_outliers = np.random.uniform(low = -4, high = 4, size = (20, 3))
X = np.r_[X + 2, X - 2, X_outliers]
```

Fit the Nearest Neighbors Model

```
k = 8 # number of neighbors
knn = NearestNeighbors(n_neighbors = k)
knn.fit(X)
```

```
▾       NearestNeighbors
NearestNeighbors(n_neighbors=8)
```

Calculate distance to k-th nearest neighbors for each point

```
distances, _ = knn.kneighbors(X)
kth_distance = distances[:, -1]
```

Set threshold for outlier detection

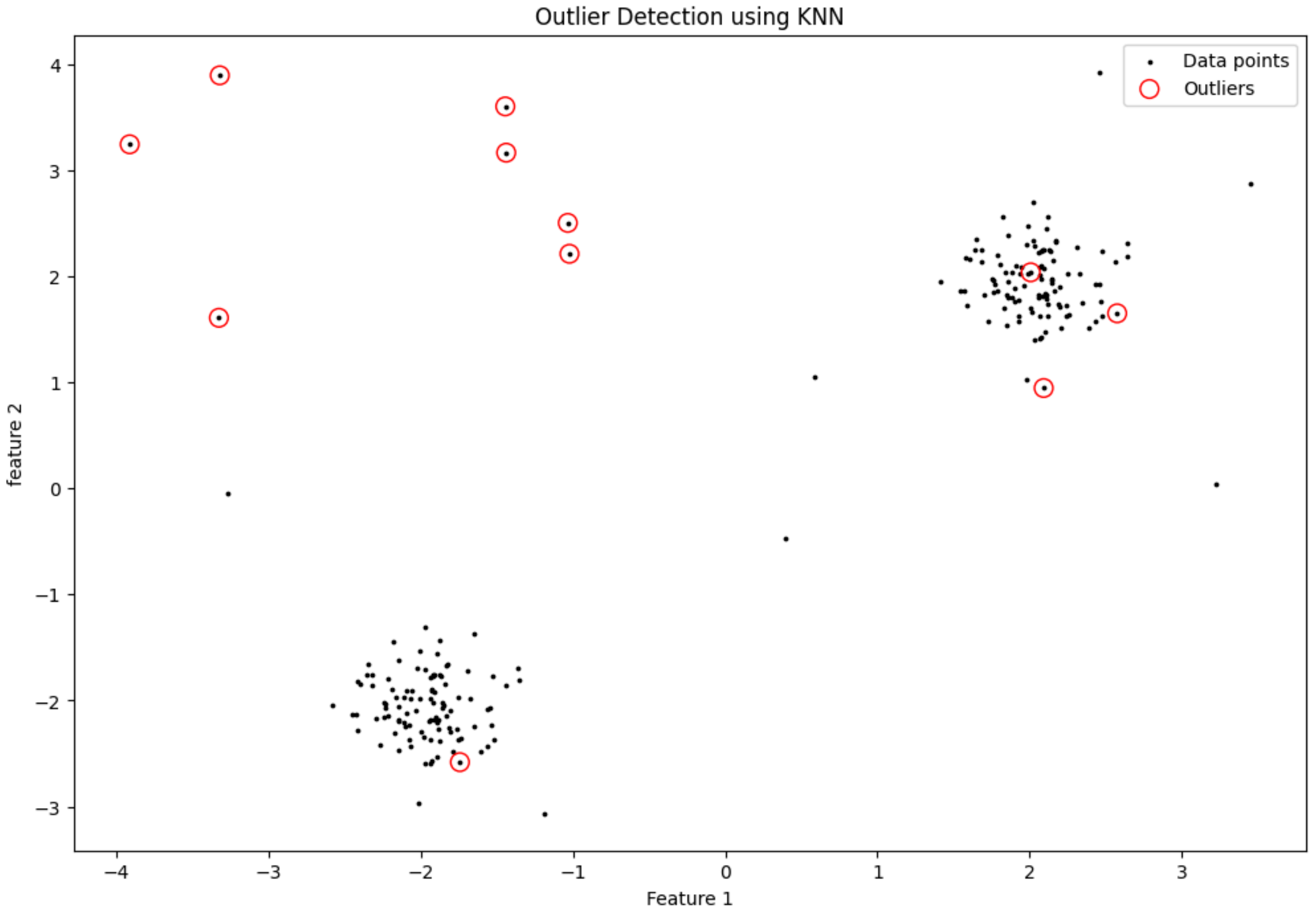
```
threshold = np.percentile(kth_distance, 95)
```

Detect outliers

```
outlier_indices = np.where(kth_distance > threshold)[0]
outliers = X[outlier_indices]
```

Plot the results

```
plt.figure(figsize = (12,8))
plt.scatter(X[:, 0], X[:, 1], color = 'k', s = 3, label = 'Data points')
plt.scatter(outliers[:, 0], outliers[:, 1], marker = 'o', edgecolors = 'r', facecolors = 'none', s =
plt.title('Outlier Detection using KNN')
plt.xlabel('Feature 1')
plt.ylabel('feature 2')
plt.legend()
plt.show()
```



Conclusion: KNN outlier detection relies on the premise that outliers have relatively few neighbors within a certain distance threshold. The algorithm involves fitting a KNN model to the data and calculating the distances to the k-nearest neighbors for each point. By setting a threshold based on these distances, points that fall beyond this threshold are considered outliers.