Pseudocode

```
Algorithm: K-Nearest Neighbors (KNN)

Input:
    X_train  (training data)
    y_train  (training labels)
    X_test   (test sample)
    k        (number of neighbors)

Output:
    Predicted class for X_test

For each test sample x in X_test:
    For each training sample xi in X_train:
        Compute distance d(x, xi)
    Sort distances in ascending order
    Select k nearest neighbors
    Take majority vote of their labels
    Assign the most frequent label to x
```

Python Code

```python
import numpy as np

def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))

def knn_predict(X_train, y_train, X_test, k=3):
    predictions = []

    for x in X_test:
        distances = []
        for i in range(len(X_train)):
            d = euclidean_distance(x, X_train[i])
            distances.append((d, y_train[i]))

        distances.sort(key=lambda x: x[0])
        k_nearest = distances[:k]

        labels = [label for _, label in k_nearest]
        predictions.append(max(set(labels), key=labels.count))
```

```
    return np.array(predictions)
```