Home | bill.csv | SVM | Untitled8 | knn ml practical | Inbox (1,391) - s22_chinta | ML_Practical-and-Project | code for implementation

localhost:8888/notebooks/knn%20ml%20practical.ipynb?

jupyter knn ml practical Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python [conda env:base] *

[1]:
```
import numpy as np
from collections import Counter

def euclidean_distance(point1, point2):
    """Calculates the Euclidean distance between two points."""
    return np.sqrt(np.sum((np.array(point1) - np.array(point2))**2))

# Example usage:
point1 = [1, 2]
point2 = [4, 6]
distance = euclidean_distance(point1, point2)
print(f"Euclidean distance between {point1} and {point2}: {distance}")
```

```
Euclidean distance between [1, 2] and [4, 6]: 5.0
```

[3]:
```
def knn_predict(X_train, y_train, test_point, k):
    """
    Predicts the class label for a new test point using the k-nearest neighbors.

    Args:
        X_train: Training data features (list of lists or numpy array).
        y_train: Training data labels (list or numpy array).
        test_point: The new data point to predict.
        k: The number of neighbors to consider.

    Returns:
        The predicted class label (string or number).
    """

    distances = []
    # Calculate distances between the test point and all training points
    for i in range(len(X_train)):
        distance = euclidean_distance(test_point, X_train[i])
        distances.append((distance, y_train[i])) # Store distance and label

    # Sort distances to get k-nearest neighbors
    distances.sort()

    # Get the labels of the k-nearest neighbors
    k_nearest_labels = [label for _, label in distances[:k]]

    # Find the most frequent label among the k-nearest neighbors (majority voting)
    most_common = Counter(k_nearest_labels).most_common(1)[0][0]
    return most_common
```


89°F Rain showers

Search

ENG IN 3:11 PM 5/27/2025

Home | bill.csv | SVM | Untitled8 | knn ml practical | Inbox (1,391) - s22_chinta | ML_Practical-and-Project | code for implementation

localhost:8888/notebooks/knn%20ml%20practical.ipynb?

jupyter knn ml practical Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python [conda env:base] *

```
    prediction = knn_predict(X_train, y_train, point, k=3)
    print(f"Test point {point} predicted label: {prediction}")
```

```
Test point [3 2] predicted label: 0
Test point [7 6] predicted label: 1
```

[3]:

```
import numpy as np
from collections import Counter

# 3D dataset (3 features)
X_train = np.array([
    [1, 2, 1],
    [2, 3, 2],
    [3, 1, 1],
    [6, 5, 4],
    [7, 7, 5],
    [8, 6, 5]
])
y_train = np.array([0, 0, 0, 1, 1, 1])

def euclidean_distance(a, b):
    return np.sqrt(np.sum((a - b)**2))

def knn_predict(X_train, y_train, x_test, k=3):
    distances = [euclidean_distance(x_test, x) for x in X_train]
    k_indices = np.argsort(distances)[:k]
    k_nearest_labels = y_train[k_indices]
    most_common = Counter(k_nearest_labels).most_common(1)
    return most_common[0][0]

X_test = np.array([
    [3, 2, 1],
    [7, 6, 4]
])

for point in X_test:
    pred = knn_predict(X_train, y_train, point, k=3)
    print(f"Test point {point} predicted label: {pred}")
```

```
Test point [3 2 1] predicted label: 0
Test point [7 6 4] predicted label: 1
```

[ ]:

Snipping Tool

Screenshot copied to clipboard
Automatically saved to screenshots folder.

Markup and share

89°F Rain showers

Search

ENG IN 3:11 PM 5/27/2025

Home | bill.csv | SVM | Untitled8 | knn ml practical | Inbox (1,391) - s22_chinta | ML_Practical-and-Project | code for implementation

localhost:8888/notebooks/knn%20ml%20practical.ipynb?

jupyter knn ml practical Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python [conda env:base] *

```
    most_common = Counter(k_nearest_labels).most_common(1)[0][0]
    return most_common

# Example usage:
#  Replace with your actual training and test data
X_train = [[1, 2], [2, 3], [3, 4], [4, 5]]
y_train = ['A', 'A', 'B', 'B']
new_point = [3, 3.5]
k = 3
prediction = knn_predict(X_train, y_train, new_point, k)
print(f"Predicted class for {new_point}: {prediction}")
```

```
Predicted class for [3, 3.5]: B
```

[1]:

```
import numpy as np
from collections import Counter

# Sample dataset (features and labels)
# Let's create a simple 2D dataset for classification
X_train = np.array([
    [1, 2],
    [2, 3],
    [3, 1],
    [6, 5],
    [7, 7],
    [8, 6]
])
y_train = np.array([0, 0, 0, 1, 1, 1])  # Labels: 0 or 1

def euclidean_distance(a, b):
    return np.sqrt(np.sum((a - b)**2))

def knn_predict(X_train, y_train, x_test, k=3):
    # Compute distances from x_test to all training points
    distances = [euclidean_distance(x_test, x) for x in X_train]

    # Get the indices of k nearest neighbors
    k_indices = np.argsort(distances)[:k]

    # Get the labels of k nearest neighbors
    k_nearest_labels = y_train[k_indices]

    # Majority vote, return the most common label
    most_common = Counter(k_nearest_labels).most_common(1)
    return most_common[0][0]

# Example test points
```


89°F Rain showers

Search

ENG IN 3:11 PM 5/27/2025

Home | bill.csv | SVM | Untitled8 | knn ml practical | Inbox (1,391) - s22_chinta | ML_Practical-and-Project | code for implementation

localhost:8888/notebooks/knn%20ml%20practical.ipynb?

jupyter knn ml practical Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

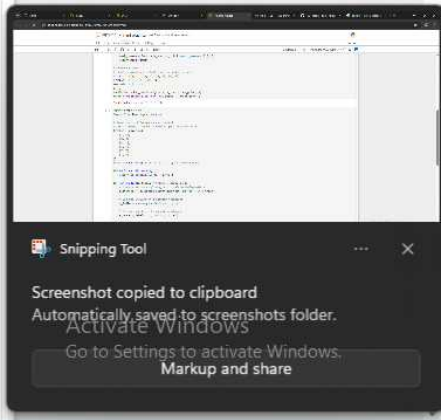
JupyterLab Python [conda env:base] *

```
    return most_common[0][0]

# Example test points
X_test = np.array([
    [3, 2],
    [7, 6]
])

for point in X_test:
    prediction = knn_predict(X_train, y_train, point, k=3)
    print(f"Test point {point} predicted label: {prediction}")
```

```
Test point [3 2] predicted label: 0
Test point [7 6] predicted label: 1
```

[3]:

```
import numpy as np
from collections import Counter

# 3D dataset (3 features)
X_train = np.array([
    [1, 2, 1],
    [2, 3, 2],
    [3, 1, 1],
    [6, 5, 4],
    [7, 7, 5],
    [8, 6, 5]
])
y_train = np.array([0, 0, 0, 1, 1, 1])

def euclidean_distance(a, b):
    return np.sqrt(np.sum((a - b)**2))

def knn_predict(X_train, y_train, x_test, k=3):
    distances = [euclidean_distance(x_test, x) for x in X_train]
    k_indices = np.argsort(distances)[:k]
    k_nearest_labels = y_train[k_indices]
    most_common = Counter(k_nearest_labels).most_common(1)
    return most_common[0][0]

X_test = np.array([
    [3, 2, 1],
    [7, 6, 4]
])

for point in X_test:
    pred = knn_predict(X_train, y_train, point, k=3)
```



Snipping Tool

Screenshot copied to clipboard
Automatically saved to screenshots folder.

Activate Windows
Go to Settings to activate Windows.
Markup and share

89°F Rain showers

Search

ENG IN

3:11 PM 5/27/2025