Manifold Learning and Isomap (Isometric Mapping)

Introduction:

Manifold learning is a set of techniques used for nonlinear dimensionality reduction. Isomap, short for Isometric Mapping, is one such manifold learning algorithm. It aims to capture the intrinsic geometry of high-dimensional data and represent it in a lower-dimensional space. Isomap, in particular, focuses on preserving the geodesic distances between all data points, providing a more accurate representation of the underlying structure compared to linear methods.

Intuition:

The intuition behind Isomap lies in the preservation of pairwise geodesic distances. While linear methods, like Principal Component Analysis (PCA), may not capture the true relationships between points in a high-dimensional space, Isomap considers the manifold's geometry. It assumes that the data lies on a low-dimensional manifold within the high-dimensional space and aims to unfold this manifold.

Algorithm:

1. **Neighborhood Graph:**
    - Construct a neighborhood graph by connecting each point to its k-nearest neighbors. This graph represents the local pairwise relationships in the data.
2. **Geodesic Distances:**
    - Compute the geodesic distances between all pairs of points on the neighborhood graph. Geodesic distance refers to the shortest path along the edges of the graph.
3. **Isomap Embedding:**
    - Apply classical multidimensional scaling (MDS) to the matrix of geodesic distances to obtain a lower-dimensional representation of the data while preserving the pairwise distances.

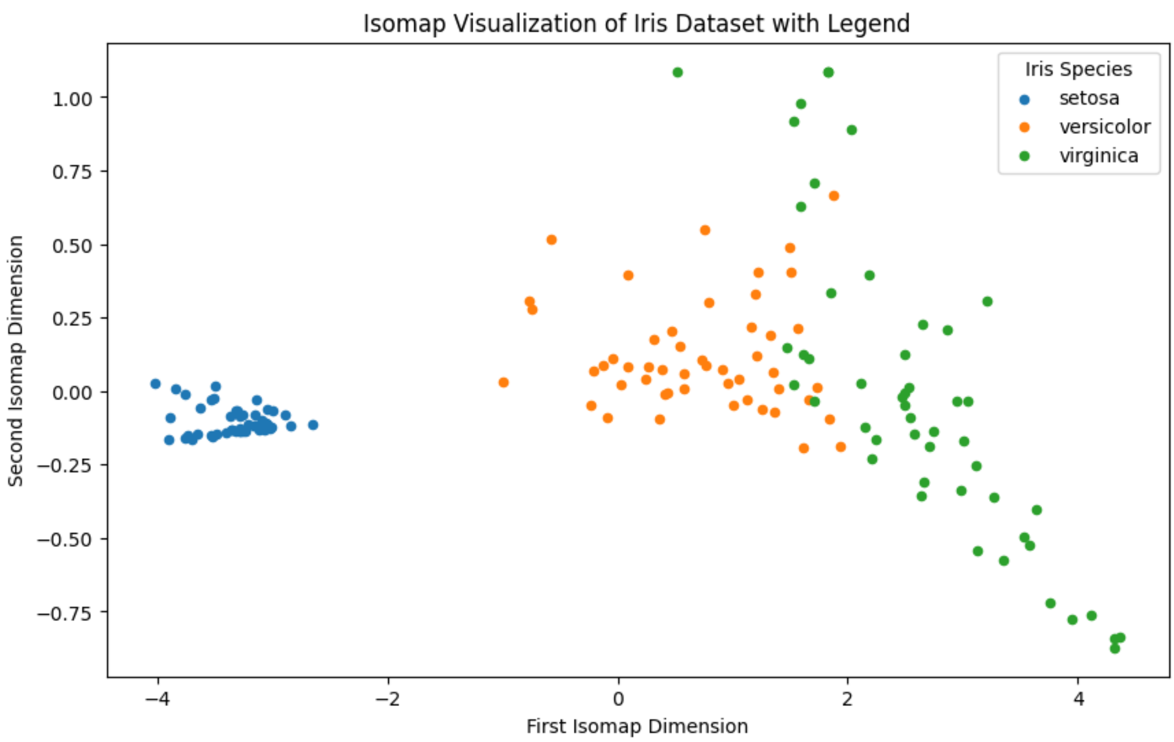
Implementation in Python:

Iris Data

Let's implement Isomap using the scikit-learn library in Python. We'll use a synthetic dataset for demonstration purposes.

```python
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.manifold import Isomap

# Load the Iris dataset
iris = datasets.load_iris()
X = iris.data
species = iris.target_names[iris.target]  # Convert target variable to species names
```

```

# Apply Isomap to the data
n_neighbors = 10
n_components = 2
isomap = Isomap(n_neighbors=n_neighbors, n_components=n_components)
X_isomap = isomap.fit_transform(X)

# Plot the results with a legend
plt.figure(figsize=(10, 6))

# Plot points for each species separately to create a legend
for s in np.unique(species):
    indices = (species == s)
    plt.scatter(X_isomap[indices, 0], X_isomap[indices, 1], label=s, s=20)

plt.title('Isomap Visualization of Iris Dataset with Legend')
plt.xlabel('First Isomap Dimension')
plt.ylabel('Second Isomap Dimension')
plt.legend(title='Iris Species', loc='upper right')
plt.show()

```



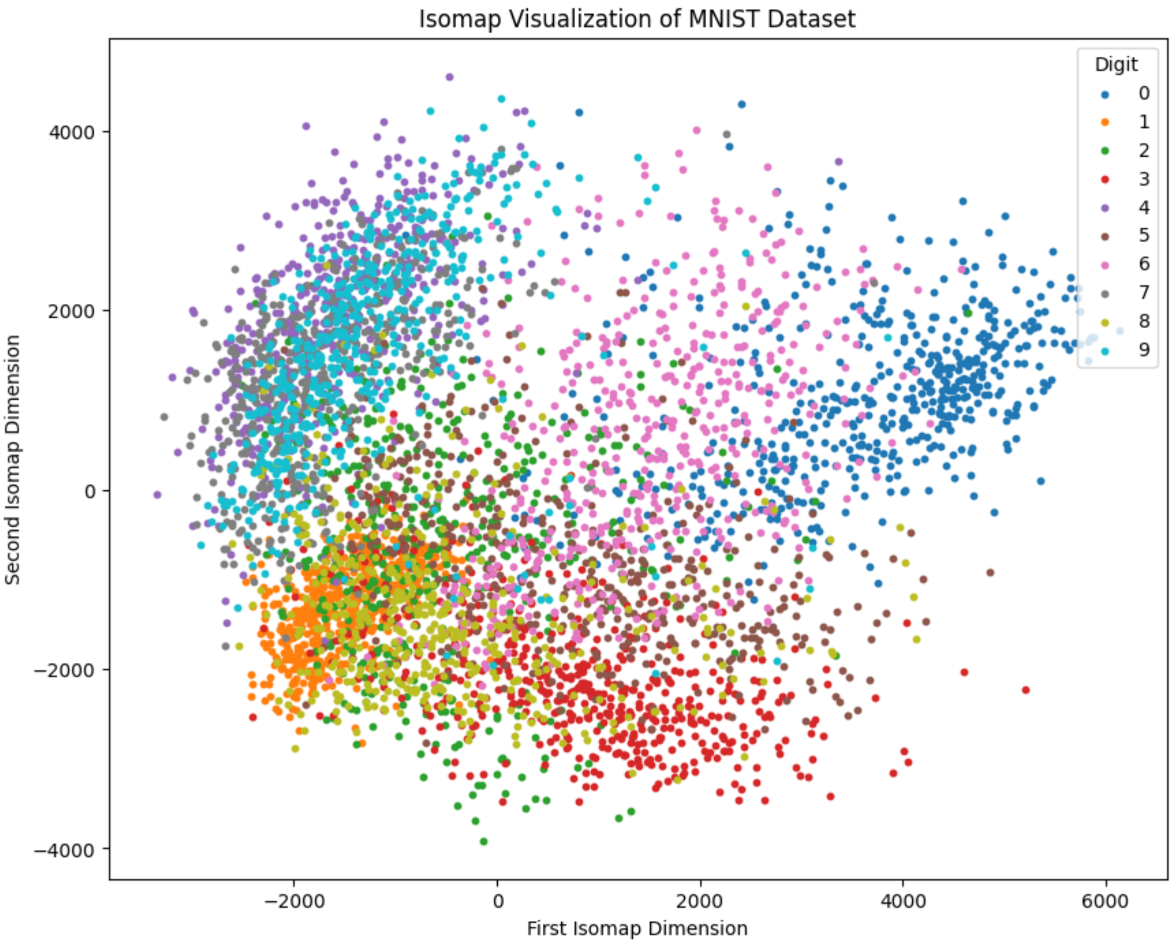
In this example:

- We generate a synthetic swiss roll dataset using scikit-learn.
- We apply Isomap to reduce the dimensionality to 2D while preserving the underlying structure.
- The result is visualized, and points are colored based on their position in the original dataset.

Replace the iris dataset with your own dataset for a real-world application of Isomap. Adjust parameters like `n_neighbors` based on the characteristics of your data.

MNIST Data

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_openml
from sklearn.manifold import Isomap

# Load MNIST dataset
mnist = fetch_openml('mnist_784')
X = mnist.data.astype('float64')
y = mnist.target.astype('int')

# Sample only a subset of the data for faster processing (optional)
sample_size = 5000
idx = np.random.choice(len(X), sample_size, replace=False)
X_sample = X.iloc[idx]  # Use iloc for integer-based indexing
y_sample = y.iloc[idx]

# Apply Isomap to the data
n_neighbors = 30
n_components = 2
isomap = Isomap(n_neighbors=n_neighbors, n_components=n_components)
X_isomap = isomap.fit_transform(X_sample)

# Plot the results
plt.figure(figsize=(10, 8))
for digit in range(10):
    plt.scatter(X_isomap[y_sample == digit, 0], X_isomap[y_sample == digit, 1], label=str(digit), s=10)
plt.title('Isomap Visualization of MNIST Dataset')
plt.xlabel('First Isomap Dimension')
plt.ylabel('Second Isomap Dimension')
plt.legend(title='Digit', loc='upper right')
plt.show()
```


Isomap Visualization of MNIST Dataset
Digit
0
1
2
3
4
5
6
7
8
9
4000
2000
0
−2000
−4000
Second Isomap Dimension
−2000
0
2000
4000
6000
First Isomap Dimension