Unsupervised Learning Metrics
Dimensionality Reduction Techniques

Dimensionality Reduction: Methods & Metrics

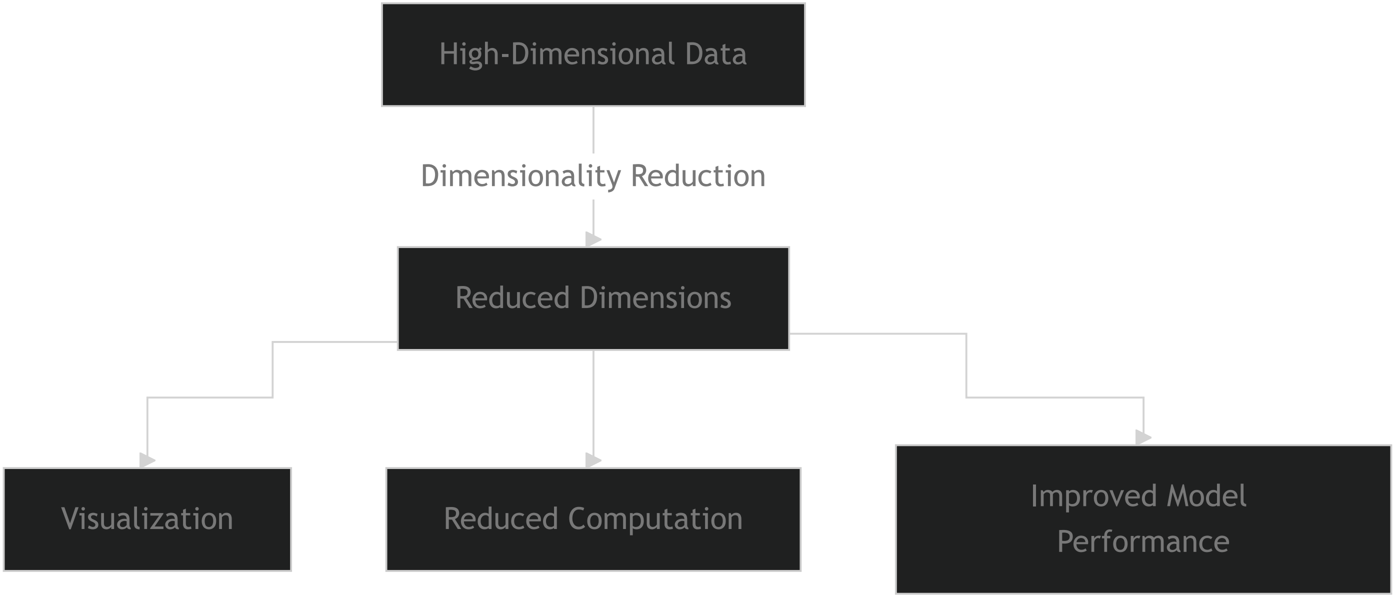
Learning Objectives

By the end of this lesson, students will be able to:

- Implement dimensionality reduction techniques (PCA, t-SNE).
- Evaluate dimensionality reduction using appropriate metrics.
- Explain practical uses of dimensionality reduction.

What is Dimensionality Reduction?

Dimensionality reduction simplifies datasets by reducing the number of features while preserving important patterns. It aids visualization, reduces computational costs, and can improve model performance.



Hands-On PCA Example

PCA (Principal Component Analysis) transforms data into fewer dimensions, emphasizing the most important information.

Copy

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

# Generate synthetic dataset
np.random.seed(42)
data = np.random.rand(100, 5)

# Apply PCA
pca = PCA(n_components=2)
reduced_data = pca.fit_transform(data)

# Explained variance
explained_variance = pca.explained_variance_ratio_.sum()
print(f"Explained Variance (2 components): {explained_variance:.2f}")

# Visualization
```

```
plt.scatter(reduced_data[:, 0], reduced_data[:, 1])
plt.title('PCA Dimensionality Reduction')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

Hands-On with t-SNE

t-SNE (t-distributed Stochastic Neighbor Embedding) is ideal for visualizing data by preserving local structures and relationships.

Example

Copy

```
from sklearn.manifold import TSNE

# Apply t-SNE
tsne = TSNE(n_components=2, random_state=42)
tsne_data = tsne.fit_transform(data)

# Visualization
plt.scatter(tsne_data[:, 0], tsne_data[:, 1], cmap='viridis')
plt.title('t-SNE Dimensionality Reduction')
plt.xlabel('t-SNE Component 1')
plt.ylabel('t-SNE Component 2')
plt.show()
```

Use Case: Particularly useful for visualizing clusters and relationships in complex datasets.

Modern Alternative: UMAP

UMAP (Uniform Manifold Approximation and Projection) is a more recent dimensionality reduction technique that addresses some limitations of t-SNE while offering additional benefits:

- **Faster computation** than t-SNE, especially for large datasets

- Better **preservation of global structure** while maintaining local relationships
- **Theoretical foundation** in manifold learning and topological data analysis
- Supports **supervised** and **semi-supervised** learning

Example

Copy

```
from umap import UMAP

# Apply UMAP
umap = UMAP(n_components=2, random_state=42)
umap_data = umap.fit_transform(data)

# Visualization
plt.scatter(umap_data[:, 0], umap_data[:, 1], cmap='viridis')
plt.title('UMAP Dimensionality Reduction')
plt.xlabel('UMAP Component 1')
plt.ylabel('UMAP Component 2')
plt.show()
```

When to Use UMAP:

- Large-scale data visualization
- When computational speed is important
- When both local and global structure preservation matter
- For creating features for downstream machine learning tasks

Additional Metrics

- **Explained Variance Ratio:** Proportion of variance explained by PCA components.
- **Reconstruction Error:** Accuracy measure often used with Autoencoders.

Reflect & Discuss

Consider your current or past projects. When might dimensionality reduction add the most value to your analysis?

< Previous

Attributions

Next >