```
Start coding or generate with AI.
```

Hour 3: Hierarchical Clustering: Unveiling Data Hierarchies

1. Introduction to Hierarchical Clustering

Hierarchical clustering is a technique that builds a hierarchy of clusters, revealing relationships between data points at different levels. Unlike k-Means, which requires a pre-defined number of clusters, hierarchical clustering creates a tree-like structure (dendrogram) that helps understand data at various granularities. This approach is particularly useful when the underlying structure of the data is hierarchical or when the number of clusters is unknown.

Key Concepts:

- **Hierarchy:** A nested sequence of clusters, where each level represents a different granularity of data grouping.
- **Dendrogram:** A visual representation of the hierarchical clustering process, showing the merging or splitting of clusters.
- **Agglomerative (Bottom-Up):** Starts with each data point as a singleton cluster and iteratively merges the closest clusters.
- **Divisive (Top-Down):** Starts with all data points in a single cluster and recursively splits clusters into smaller ones.
- **Proximity Matrix (Distance/Similarity Matrix):** A matrix D where D_{ij} represents the distance or dissimilarity between data points i and j

Why is this important? It's useful for exploring data structures at multiple levels and discovering natural groupings.

Mathematical Foundation:

- **Distance Metrics:** Common distance metrics include Euclidean distance ($d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$), Manhattan distance ($d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$), and Minkowski distance ($d(x, y) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}$).
- **Similarity Metrics:** Cosine similarity ($s(x, y) = \frac{x \cdot y}{||x|| ||y||}$) is often used for text or high-dimensional data.

Assignment: Learn about Mahalanobish Distance and write a short report on it.

Hierarchical clustering aims to build a hierarchy of clusters, represented as a tree-like structure called a dendrogram. This approach is particularly useful when the underlying structure of the data is hierarchical or when the number of clusters is unknown.

Key Concepts:

- **Hierarchy:** A nested sequence of clusters, where each level represents a different granularity of data grouping.
- **Dendrogram:** A visual representation of the hierarchical clustering process, showing the merging or splitting of clusters.
- **Agglomerative (Bottom-Up):** Starts with each data point as a singleton cluster and iteratively merges the closest clusters.
- **Divisive (Top-Down):** Starts with all data points in a single cluster and recursively splits clusters into smaller ones.
- **Proximity Matrix (Distance/Similarity Matrix):** A matrix D where D_{ij} represents the distance or dissimilarity between data points i and j.

Mathematical Foundation:

- **Distance Metrics:** Common distance metrics include Euclidean distance ($d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$), Manhattan distance ($d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$), and Minkowski distance ($d(x, y) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}$).
- **Similarity Metrics:** Cosine similarity ($s(x, y) = \frac{x \cdot y}{||x|| ||y||}$) is often used for text or high-dimensional data.

2. Agglomerative Hierarchical Clustering

Agglomerative clustering is the most common type. It starts with each data point as a separate cluster and iteratively merges the closest pairs of clusters until all points belong to a single cluster.

Steps:

1. **Initialization:** Each data point is considered a single cluster.
2. **Distance Calculation:** Calculate the distance between all pairs of clusters.
3. **Merging:** Merge the two closest clusters.
4. **Update Distance Matrix:** Recalculate distances between the new cluster and the remaining clusters.
5. **Repeat:** Repeat steps 2-4 until all data points are in a single cluster.

Linkage Methods:

- **Single Linkage:** Minimum distance between points in two clusters. Tends to form long, chain-like clusters.
- **Complete Linkage:** Maximum distance between points in two clusters. Tends to form compact clusters.
- **Average Linkage:** Average distance between points in two clusters. A compromise between single and complete linkage.
- **Ward's Linkage:** Minimizes the variance within clusters. Tends to form equally sized, compact clusters.

Example:

Imagine clustering cities based on their geographical proximity. Single linkage might connect cities in a long chain, while complete linkage would form compact groups of nearby cities.

2.1. Agglomerative Hierarchical Clustering (Mathematically)

Agglomerative clustering builds the hierarchy by iteratively merging the closest clusters. The choice of linkage method determines how the distance between clusters is defined.

Steps:

1. **Initialization:** Each data point forms a singleton cluster.
2. **Distance Calculation:** Compute the pairwise distance matrix D between all clusters.
3. **Merging:** Merge the two clusters C_i and C_j with the minimum distance D_{ij}.
4. **Update Distance Matrix:** Recalculate D based on the new cluster.
5. **Repeat:** Repeat steps 3-4 until all data points belong to a single cluster.

Linkage Methods (Mathematical Definitions):

- **Single Linkage (Nearest Neighbor):** $D(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$
- **Complete Linkage (Furthest Neighbor):** $D(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y)$
- **Average Linkage:** $D(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i, y \in C_j} d(x, y)$
- **Ward's Linkage:** Minimizes the total within-cluster variance. Mathematically, it aims to minimize the increase in the squared error sum when merging clusters.

Computational Complexity:

- The time complexity of agglomerative clustering is $O(n^3)$ in the naive implementation, where n is the number of data points.

- Using efficient data structures like priority queues, the complexity can be reduced to $O(n^2 \log n)$.

3. Dendrogram Interpretation

A dendrogram is a tree-like diagram that visually represents the hierarchy of clusters. The y-axis represents the distance (or similarity) between clusters. The x-axis represents the data points.

How to Interpret:

- **Height:** The height at which two clusters are joined indicates the distance between them. Higher joins mean less similar clusters. In other words; The height of a join represents the distance between the merged clusters.
- **Number of Clusters:** To choose the number of clusters, draw a horizontal line at a threshold distance. By drawing a horizontal line at a specific height, the number of intersections with vertical lines indicates the number of clusters.
- **Cophenetic Correlation Coefficient:** Measures how well the dendrogram preserves the pairwise distances. It is defined as the correlation between the cophenetic distances (distances in the dendrogram) and the original distances. A high cophenetic correlation (close to 1) indicates a good representation of the data's structure.

Mathematical Definition:

- Cophenetic Correlation Coefficient: $c = \frac{\sum_{i<j}(x_{ij}-\bar{x})(y_{ij}-\bar{y})}{\sqrt{\sum_{i<j}(x_{ij}-\bar{x})^2 \sum_{i<j}(y_{ij}-\bar{y})^2}}$, where x_{ij} are original distances, and y_{ij} are cophenetic distances.

Example:

If a dendrogram shows a join at a high distance, it means those clusters are quite dissimilar. If we draw a horizontal line that intersects three vertical lines, we can conclude that there are three clusters in the data.

4. Comparing k-Means and Hierarchical Clustering

k-Means:

- Requires pre-defined number of clusters (k).
- Efficient for large datasets.
- Produces flat clusters.
- Sensitive to initial centroids.

Hierarchical Clustering:

- Does not require pre-defined number of clusters.
- Computationally expensive for large datasets.
- Produces a hierarchy of clusters.
- Provides insights into data relationships.

When to use which:

- Use k-Means when you know the number of clusters and have a large dataset.
- Use hierarchical clustering when you want to explore data at multiple levels and visualize relationships.

Use the following table to compare between the both.

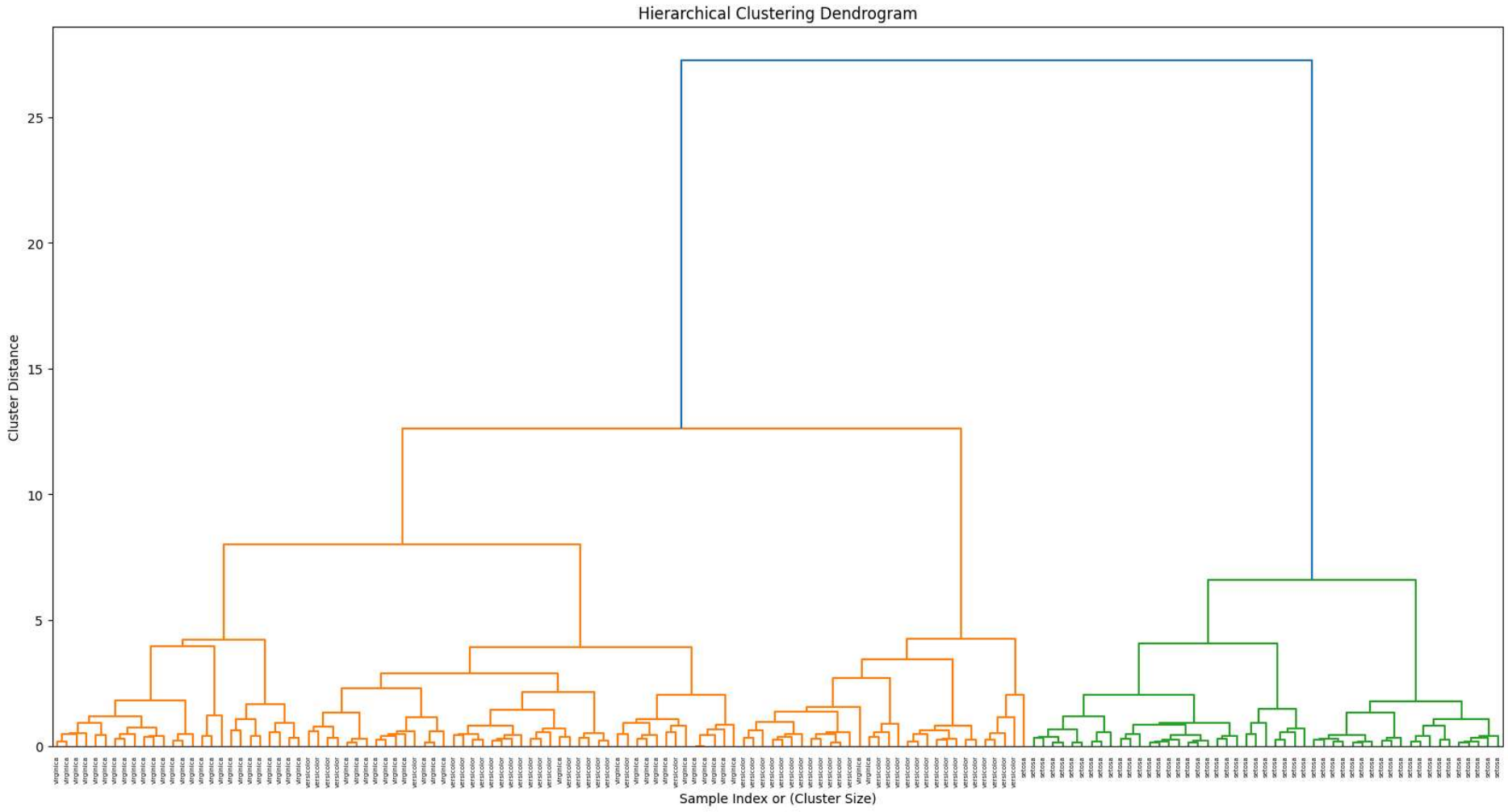
Feature	k-Means	Hierarchical Clustering
Number of Clusters	Requires pre-defined k	Does not require pre-defined number of clusters
Computational Complexity	$O(nki)$, where n is datapoints, k clusters, i iterations	$O(n^3)$ or $O(n^2 \log n)$
Cluster Shape	Tends to produce spherical clusters	Can handle various cluster shapes
Output	Flat clusters	Hierarchy of clusters (dendrogram)
Interpretability	Less interpretable in terms of relationships	More interpretable in terms of hierarchical relationships
Sensitivity to Outliers	Sensitive to outliers	Less sensitive to outliers
Initialization	Sensitive to initial centroids	Deterministic (for agglomerative)

5. Python Implementation using Scikit-Learn

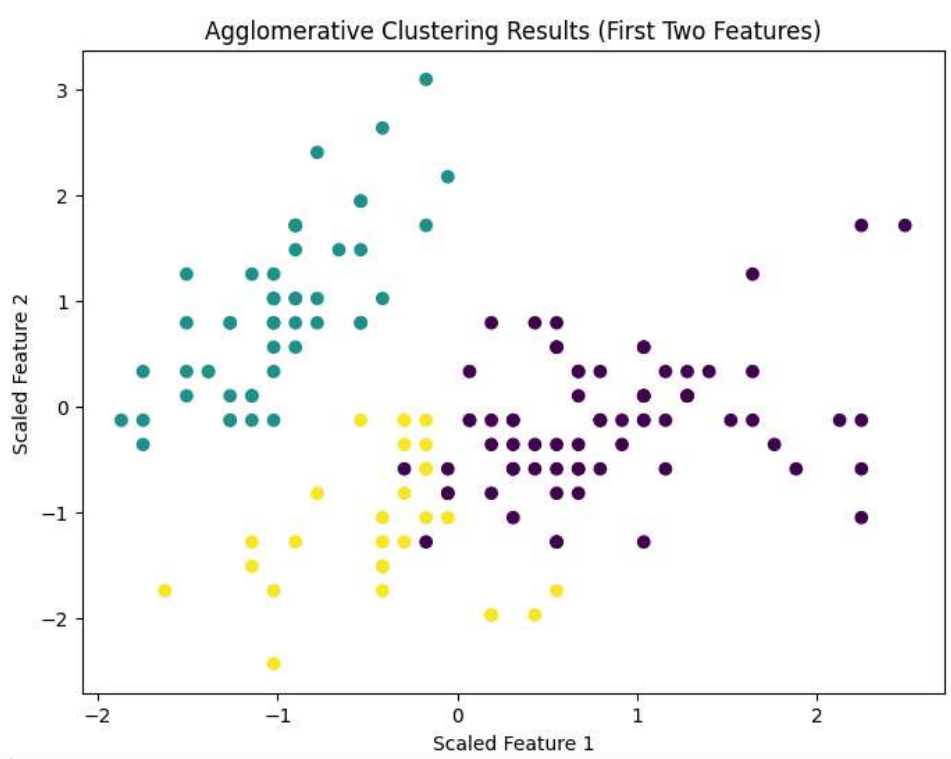
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import AgglomerativeClustering
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from scipy.cluster.hierarchy import dendrogram, linkage

# Load the Iris dataset
iris = load_iris()
X = iris.data

# Scale the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Agglomerative Clustering
agg_cluster = AgglomerativeClustering(n_clusters=3, linkage='ward')
labels = agg_cluster.fit_predict(X_scaled)

print("Cluster Labels:", labels)
```

```
Cluster Labels: [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 2 1 1 1 1 1 1 1 1 0 0 0 2 0 2 0 2 0 2 2 0 2 0 2 0 2 2 2 2 0 0 0 0
 0 0 0 0 0 2 2 2 2 0 2 0 0 2 2 2 2 0 2 2 2 2 2 0 2 2 0 0 0 0 0 0 2 0 0 0 0
 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0]
```

```
# Dendrogram Visualization
linked = linkage(X_scaled, 'ward')
plt.figure(figsize=(20, 10))
dendrogram(linked, orientation='top', labels=iris.target_names[iris.target], distance_sort='descending', show_leaf_counts=True)
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Sample Index or (Cluster Size)')
plt.ylabel('Cluster Distance')
plt.show()
```




```
# Visualize Clusters (using first two features for simplicity)
plt.figure(figsize=(8, 6))
plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=labels, cmap='viridis')
plt.title('Agglomerative Clustering Results (First Two Features)')
plt.xlabel('Scaled Feature 1')
plt.ylabel('Scaled Feature 2')
plt.show()
```



6. Practical Considerations and Limitations

- **Dataset Size:** Hierarchical clustering can be computationally expensive for large datasets.
- **Categorical Data:** Handling categorical data requires appropriate distance metrics (e.g., Gower's distance).
- **Linkage Method Selection:** The choice of linkage method can significantly impact the clustering results.
- **Sensitivity to Noise and Outliers:** Noise and outliers can distort the hierarchical structure.

7. Comparison with Other Clustering Algorithms

- **DBSCAN:** Density-based clustering, robust to outliers, can find clusters of arbitrary shapes.
- **Gaussian Mixture Models (GMMs):** Model-based clustering, assumes data is generated from a mixture of Gaussian distributions.

8. Real-World Applications

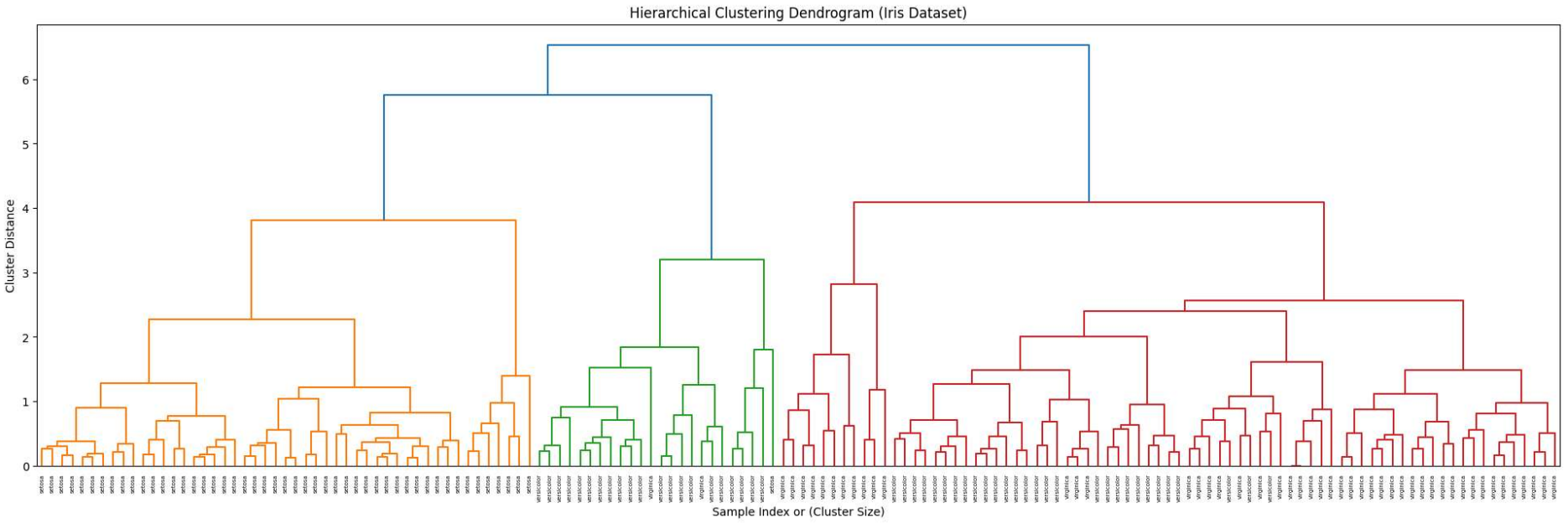
- **Genetics:** Clustering genes or individuals based on genetic similarity.
- **Document Clustering:** Grouping documents based on topic similarity.
- **Image Segmentation:** Segmenting images into regions based on pixel similarity.
- **Social Network Analysis:** Identifying communities or groups of users.

9. Interactive Exercises and Discussion Points

- **Discussion:**
  - "When would you prefer single linkage over complete linkage?"
  - "How does the choice of distance metric affect the clustering results?"
  - "What are the limitations of using dendrograms for large datasets?"

- **Exercises:**
  - Manually construct a small dendrogram.
  - Experiment with different linkage methods and observe the changes in the resulting clusters.

```
# Import required libraries
from scipy.cluster.hierarchy import linkage, dendrogram, cut_tree
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler

# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# Scale the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Use different distance metrics
linked_manhattan = linkage(X_scaled, method='complete', metric='euclidean')

# Prune dendrogram
clusters = cut_tree(linked_manhattan, n_clusters=3).reshape(-1,)

# Save dendrogram
plt.figure(figsize=(24, 7))
dendrogram(linked_manhattan, orientation='top', labels=np.array(iris.target_names)[y],
           distance_sort='descending', show_leaf_counts=True)
plt.title('Hierarchical Clustering Dendrogram (Iris Dataset)')
plt.xlabel('Sample Index or (Cluster Size)')
plt.ylabel('Cluster Distance')
plt.savefig("dendrogram.svg")
plt.show()
plt.close()
```




```
Start coding or generate with AI.
```