**Unsupervised Learning**

* In unsupervised learning, the data is **unlabeled** (no predefined output).
* The model tries to find **hidden patterns, structures, or groupings** in the data.
* **Dimensionality reduction :** Suppose you have a very large dataset with hundreds offeatures. Unsupervised learning can reduce it into fewer, more meaningful features without losing much information.

**🔹 Clustering in Unsupervised Learning**

Clustering is a **technique in unsupervised learning** where the algorithm **groups data points** into clusters (groups) based on how similar they are to each other — **without using labels (outputs)**.

**🔹 Key Idea**

* You give the algorithm **only input features (X)**, not labels (y).
* The algorithm finds **patterns or natural groupings** in the data.
* Each group (cluster) contains data points that are **more similar to each other** than to points in other groups.

**🔹 Example**

Imagine you have data about people:

* Height
* Weight
* Spending habits

You don’t tell the model if a person is "athlete", "student", or "businessman".

A clustering algorithm (like **K-Means**) might automatically form groups like:

* Cluster 1 → tall, muscular people (athletes)
* Cluster 2 → young people with moderate spending (students)
* Cluster 3 → high-spending older people (businessmen)

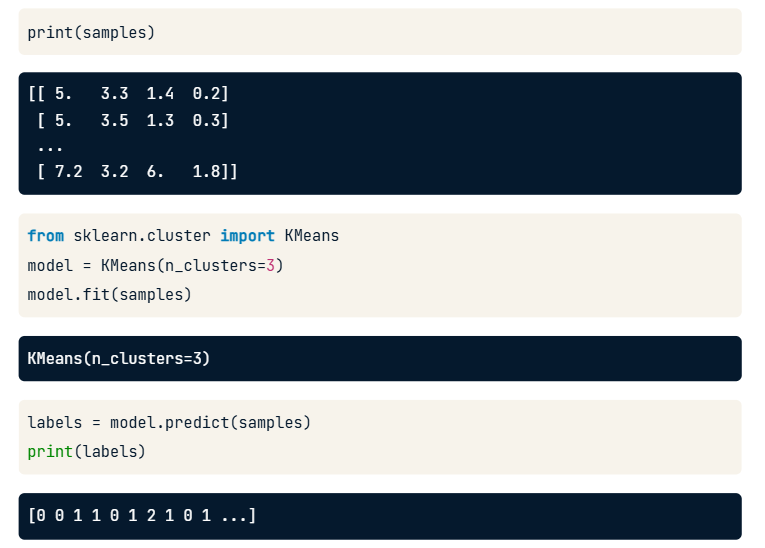
👉 Notice: The algorithm **discovered groups** without being told the labels.

**🔹 Common Clustering Algorithms**

1. **K-Means** → divides data into *k* clusters by minimizing distance within each cluster.
2. **Hierarchical Clustering** → builds a tree of clusters.
3. **DBSCAN** → groups points based on density (useful for irregular-shaped clusters).

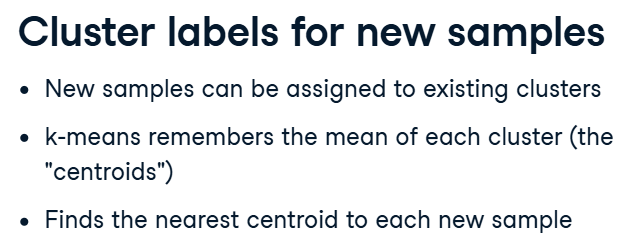
**🔹 Why is it useful?**

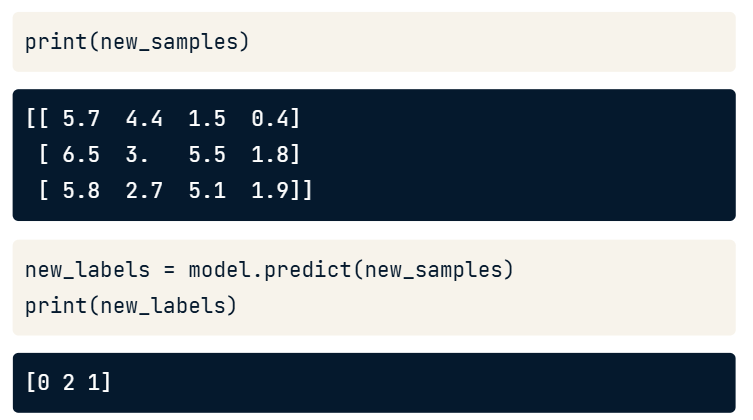
* **Market segmentation** (grouping customers by purchasing behavior)
* **Image compression** (similar pixels grouped together)
* **Anomaly detection** (points not belonging to any cluster)
* **Biology** (grouping genes/proteins with similar properties)

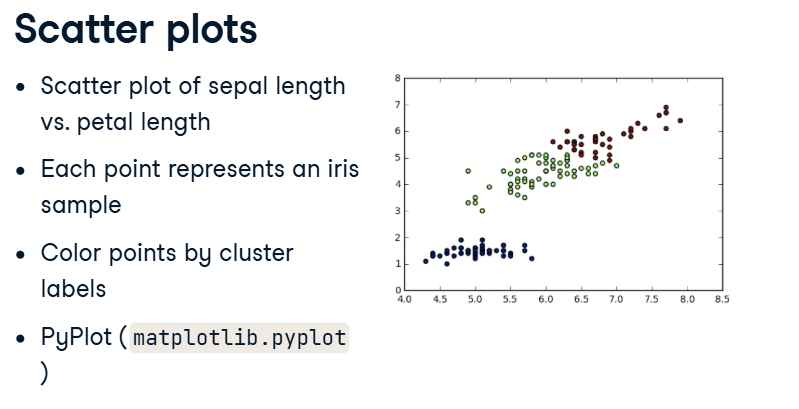


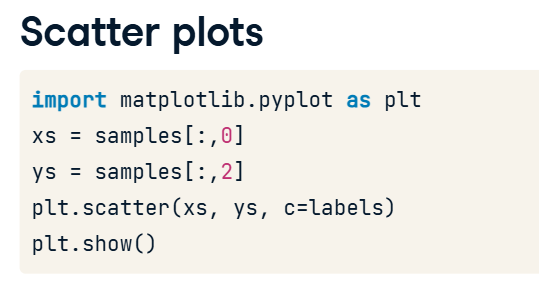
What happens internally:

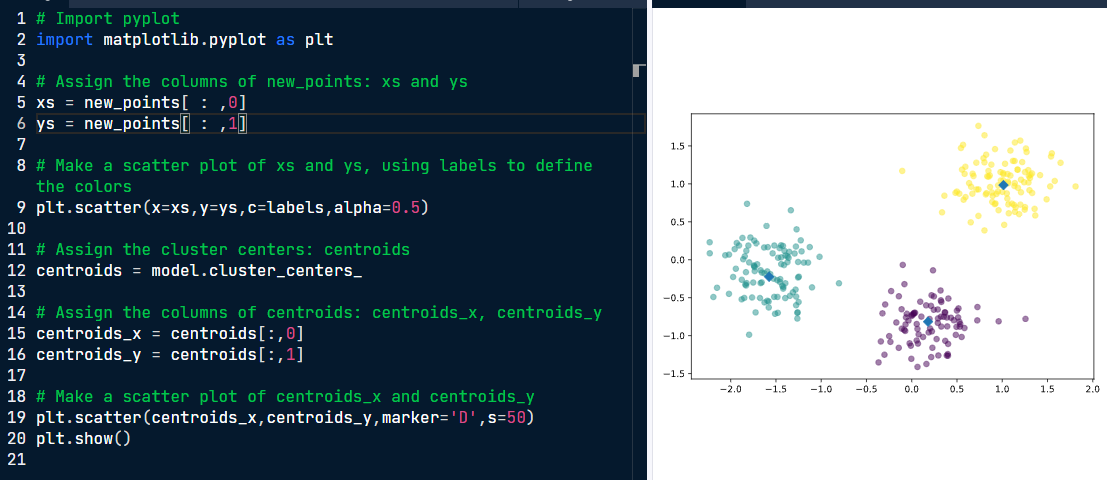
1. KMeans randomly picks **3 initial centroids** (one for each cluster).
2. Each flower (data point) is assigned to the **nearest centroid** → forms temporary clusters.
3. The algorithm recomputes the **centroid (average point)** of each cluster.
4. Steps 2–3 repeat until the centroids stop moving much (convergence).

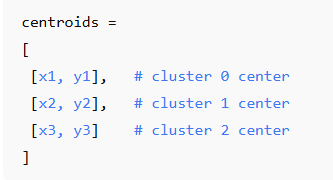




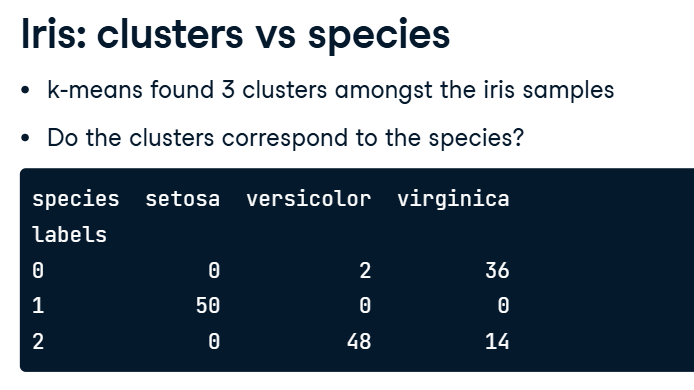


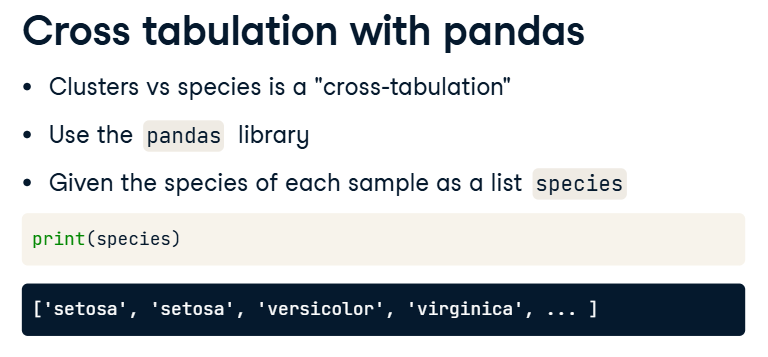




Here,



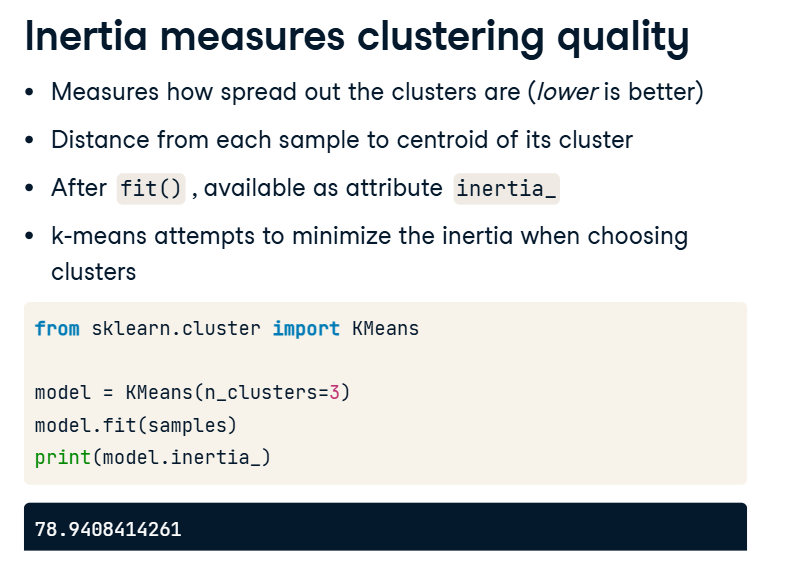


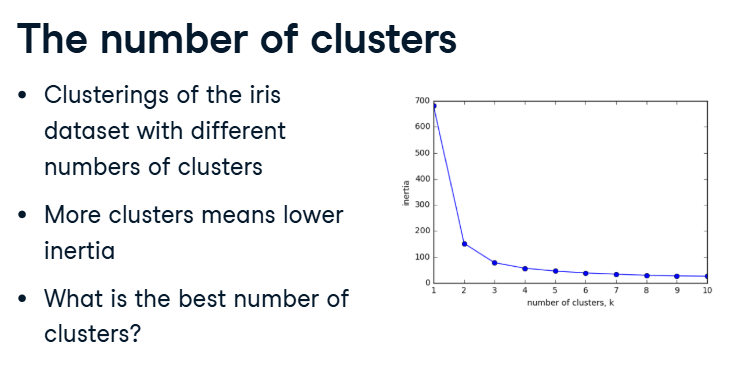


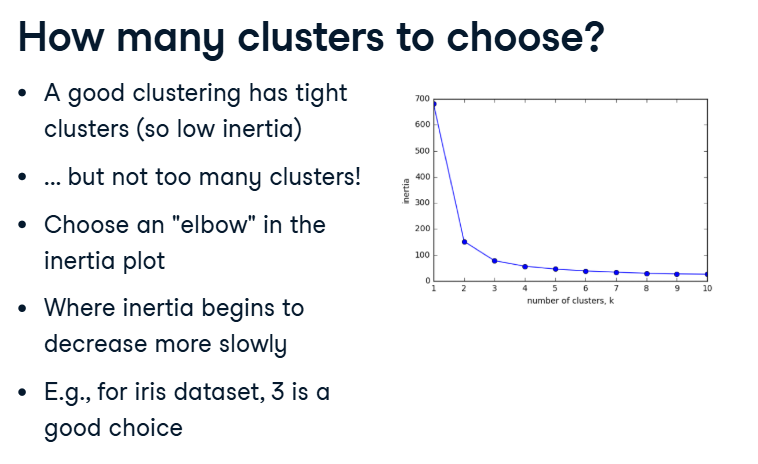


**Measuring Cluster Quality**

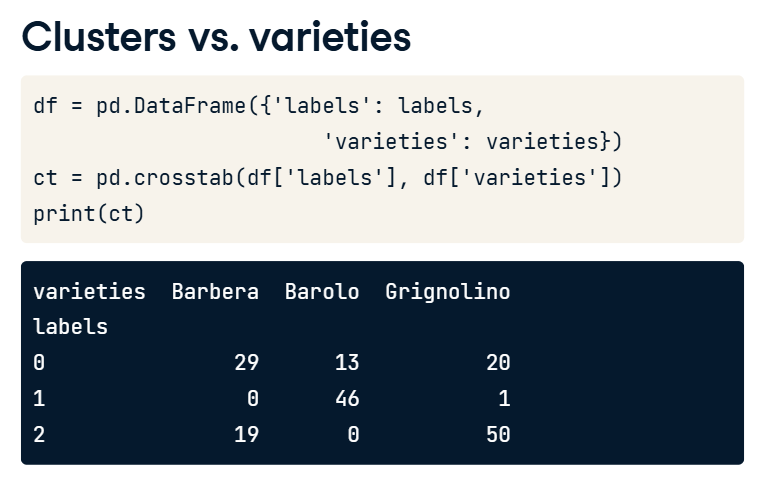
* Using only samples and their cluster labels.
* A good clustering has tight clusters and not too many clusters.
* Samples in each cluster bunched together.





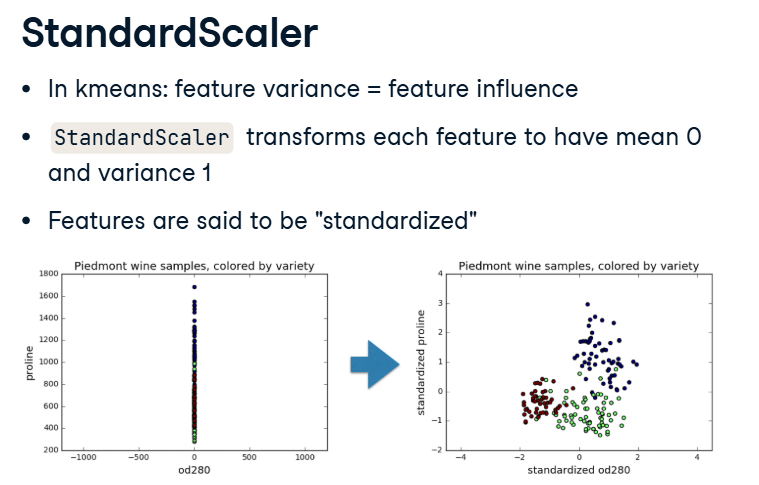


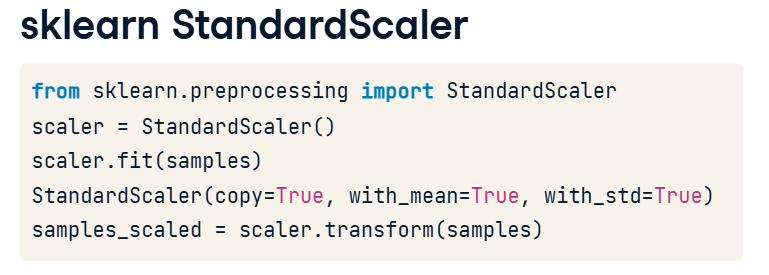
For a wine dataset,

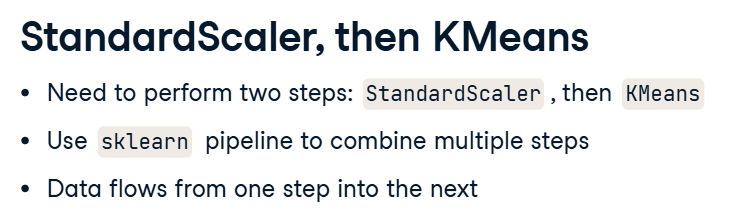


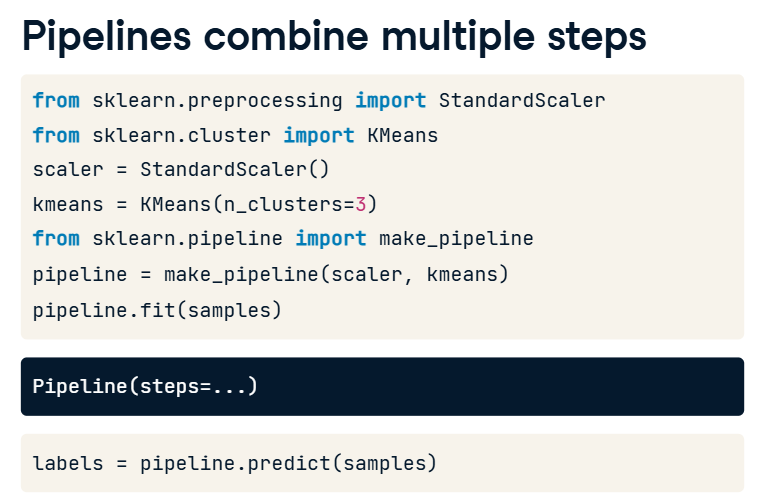
K Means Clustering doesn’t work out well for wine varieties.

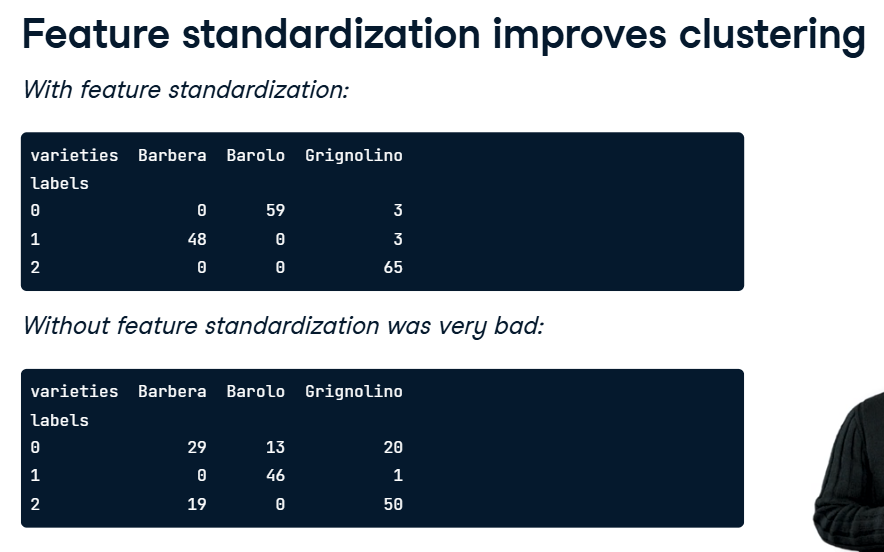


StandardScaler is a preprocessing step.









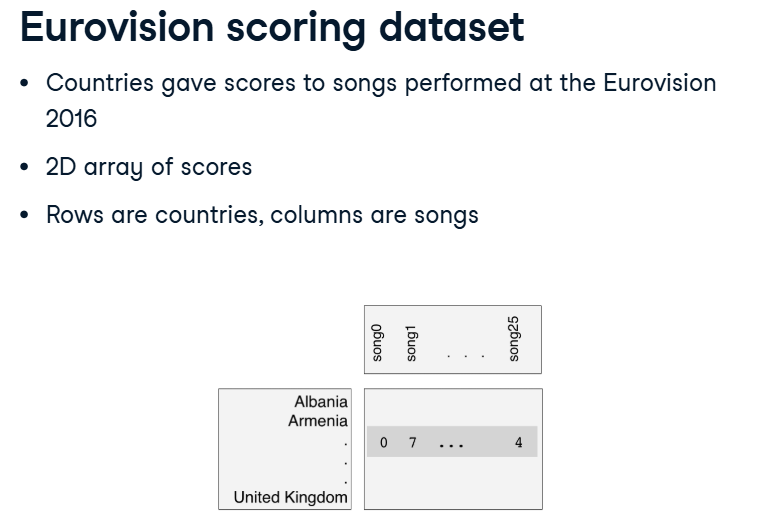
**Hierarchial Clustering**

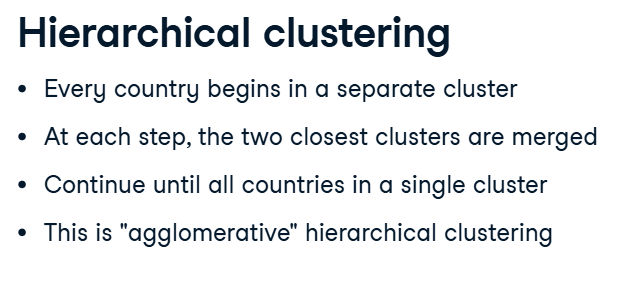
It is a type of **unsupervised learning** where we group data points into clusters based on similarity, but instead of deciding the number of clusters in advance (like in KMeans), we build a **hierarchy of clusters**.

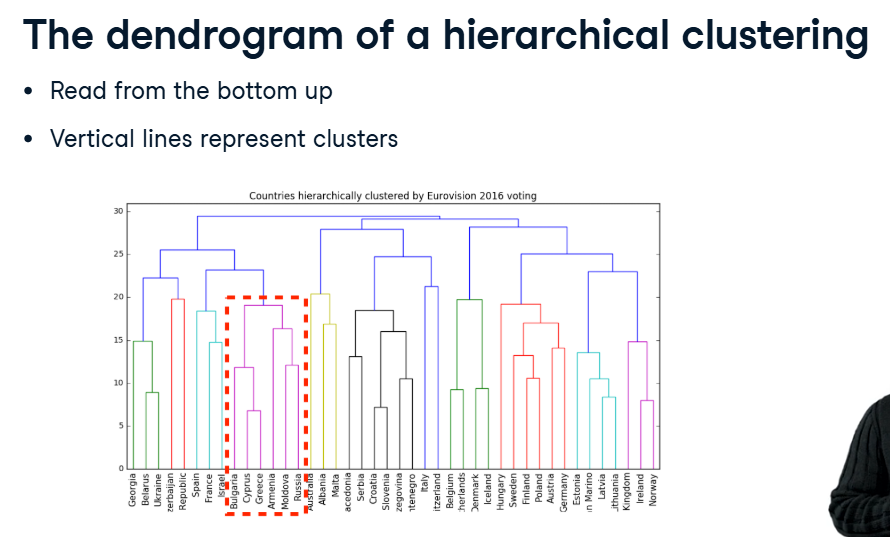


**Types of Hierarchical Clustering:**

1. **Agglomerative (Bottom-Up, most common)**
   * Start with each point as its **own cluster**.
   * Repeatedly merge the two closest clusters.
   * Continue until everything is one big cluster.
2. **Divisive (Top-Down)**
   * Start with **all points in one cluster**.
   * Repeatedly split clusters until each point is separate.







**🔹 How it works (step by step):**

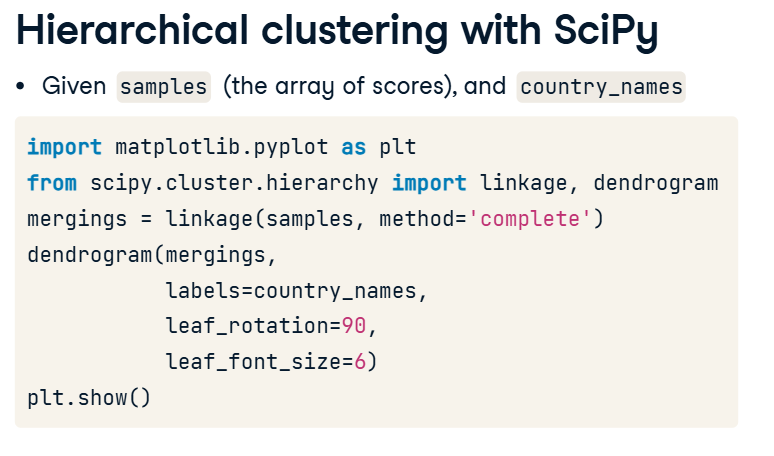
1. **Start with all countries as individual clusters**
   * At the very bottom, each country is its own cluster.
2. **Find the two most similar countries**
   * Similarity is based on some measure (like Euclidean distance, correlation, or voting similarity here).
   * Those two countries are joined together with a line.
3. **Keep merging the closest clusters**
   * After the first merge, you now have fewer clusters.
   * Find the next closest pair of clusters (could be single countries or already-merged groups) and merge them.
4. **Repeat until everything is one big cluster**
   * At the very top, you see all countries merged into one giant cluster.

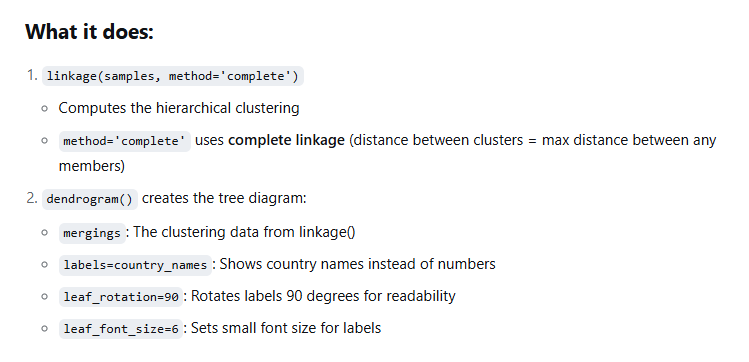
**🔹 How to read the dendrogram:**

* The **x-axis** = the countries.
* The **y-axis** = distance (or dissimilarity) between merged clusters.
* The **height of the line** where two clusters join tells you **how different they were**.
  + Low line → very similar (close relationship).
  + High line → less similar (more distant).

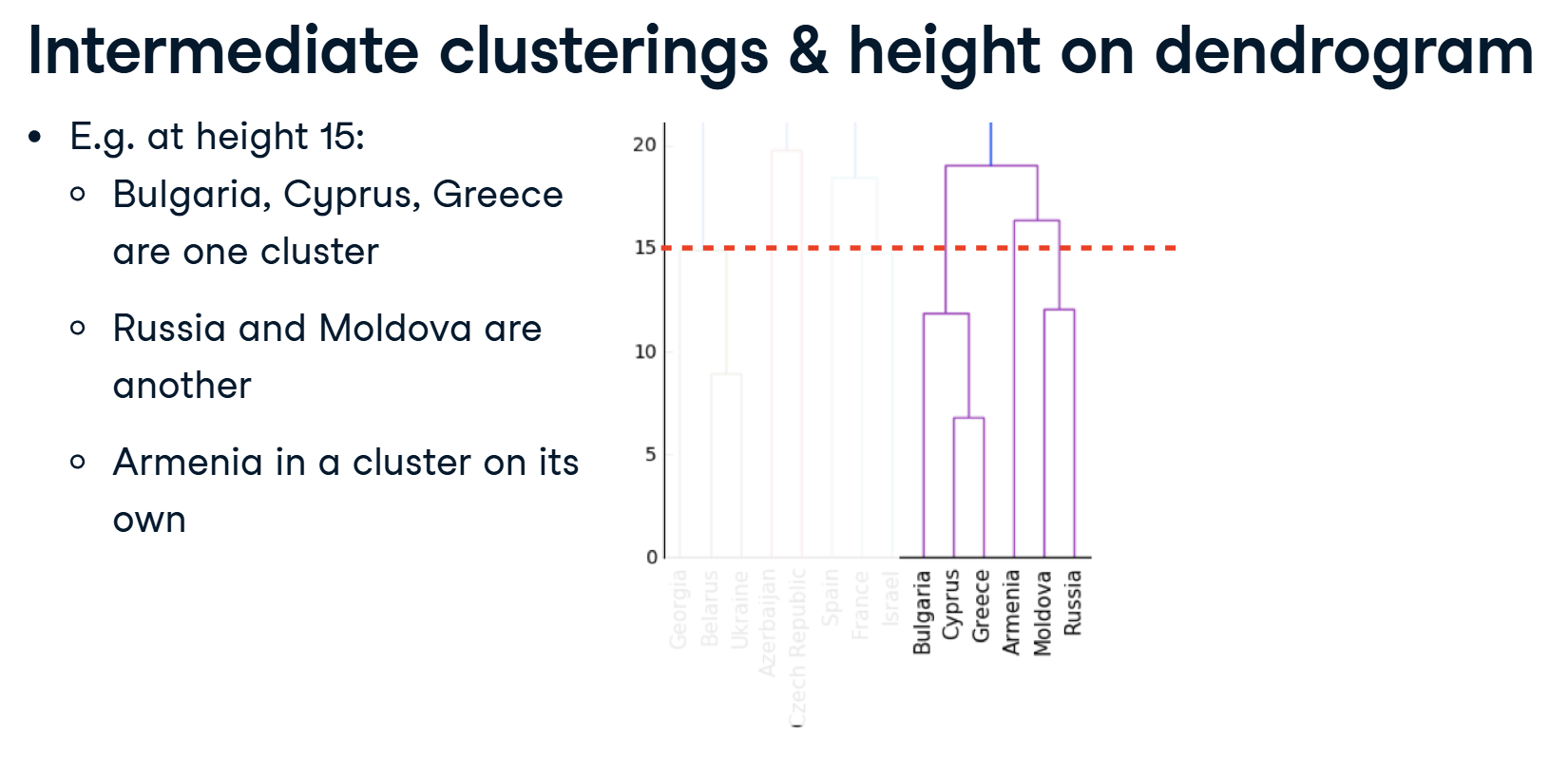
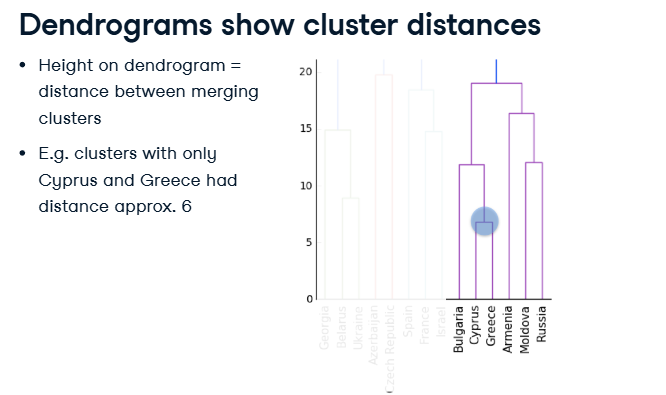
For example:

* If two countries are joined at a low height, they voted very similarly.
* If two countries are only connected way up at the top, they are not very similar.

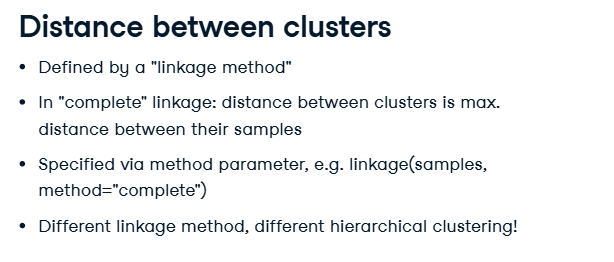


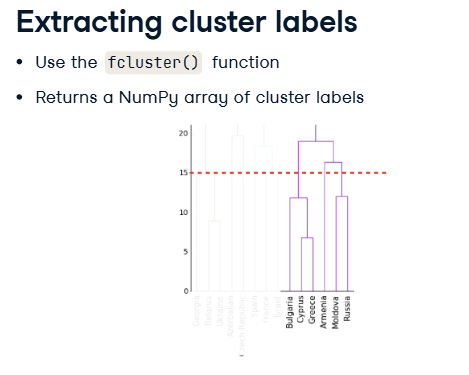
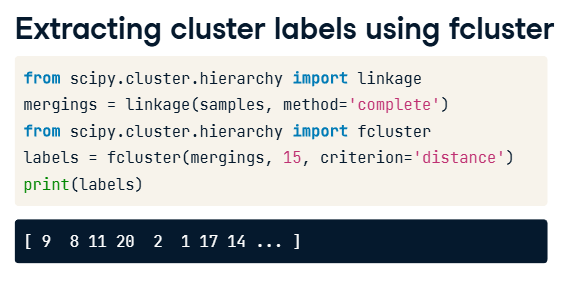


**Note: With ‘n’ data samples , there will be ‘n-1’ merge operations.**

****

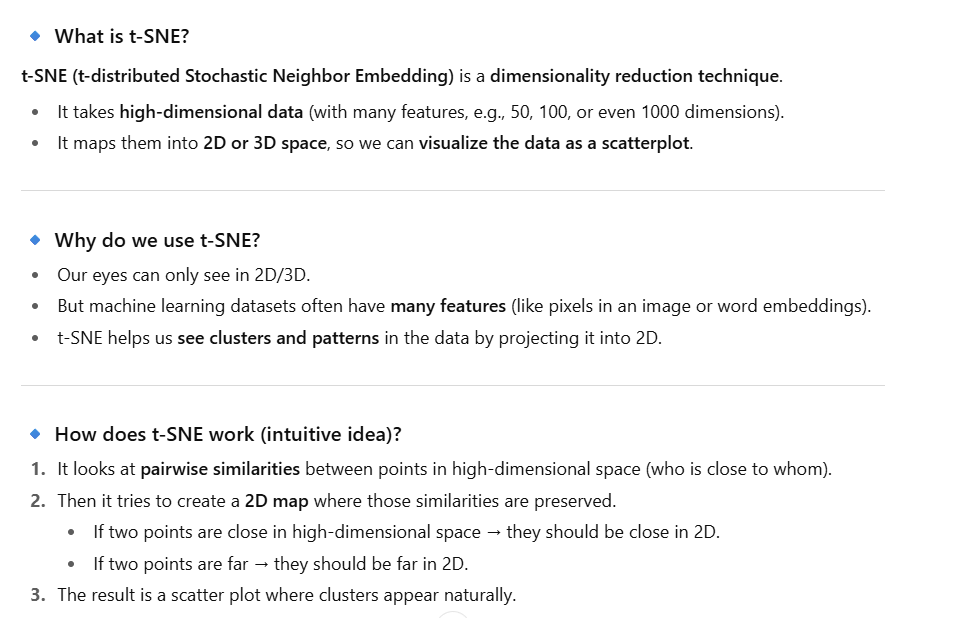
****

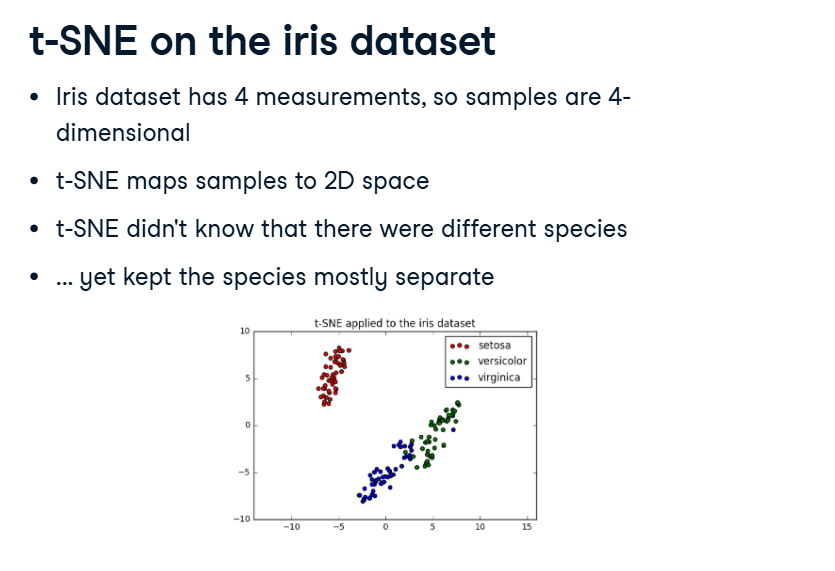
****

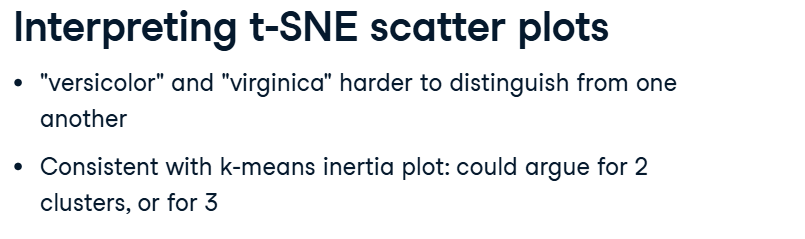
****

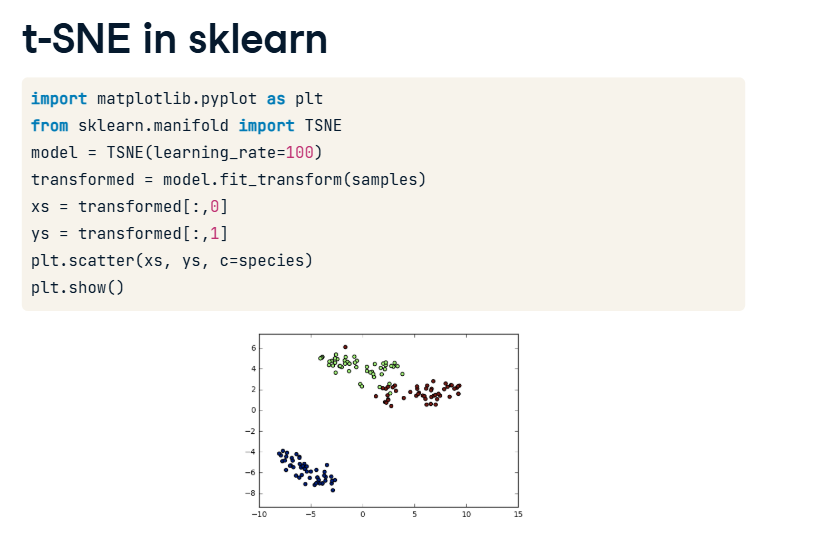
****

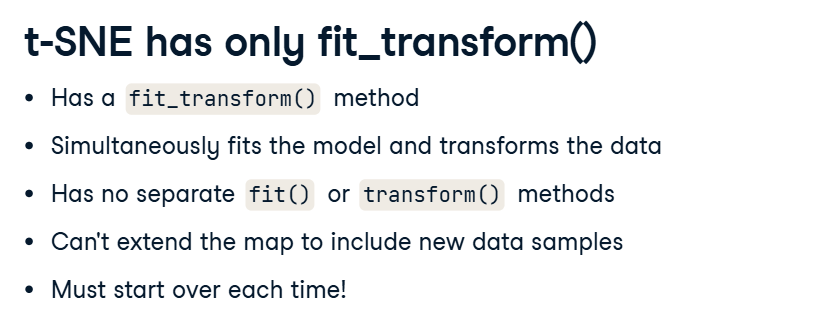
**t-SNE for 2-dimensional maps**



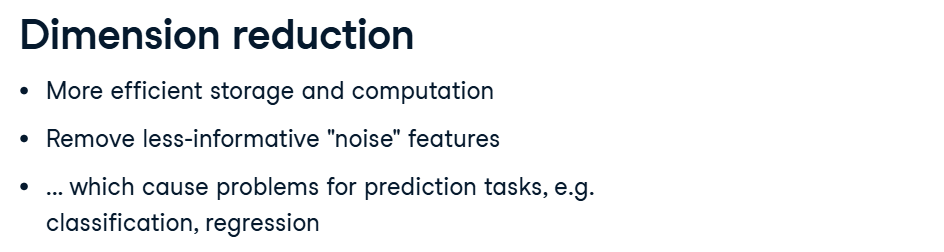












**🔹 What is PCA?**

**Principal Component Analysis (PCA)** is a **dimensionality reduction technique**.

* It takes data with **many features (dimensions)** and reduces it to fewer dimensions.
* While reducing, it tries to **keep as much important information (variance) as possible**.

**🔹 Why do we use PCA?**

* Many datasets have **lots of correlated features** (e.g., height & weight, pixels in images).
* High dimensions are harder to visualize and process.
* PCA helps by:
  + Removing redundancy (correlation).
  + Making the data easier to analyze, visualize, or feed into machine learning models.

**🔹 How does PCA work (intuitive)?**

1. **Find directions of maximum variance in the data**
   * Think of data points scattered in space.
   * PCA finds a new axis (called a **principal component**) that best captures the spread (variance) of data.
2. **Re-orient the coordinate system**
   * It rotates the original axes into new ones (principal components).
   * These new axes are **uncorrelated** and ordered by importance (1st PC captures the most variance, 2nd PC the next most, etc.).
3. **Drop less important components**
   * Keep only the first few principal components (say 2 or 3).
   * This reduces dimensionality while keeping most information.

**🔹 Example**

Suppose you have a dataset with 100 features (columns).

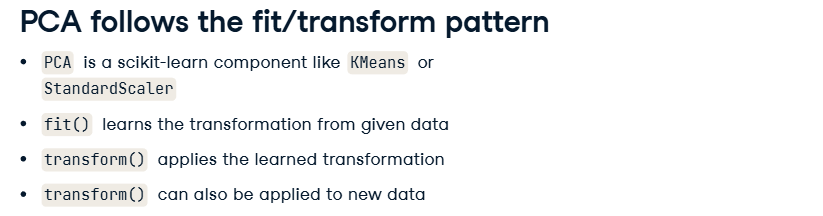
* PCA might reduce it to **2 or 3 features** that still capture **90–95% of the variation**.
* Then you can **plot it in 2D/3D** and also train ML models faster.

**🔹 Difference from t-SNE**

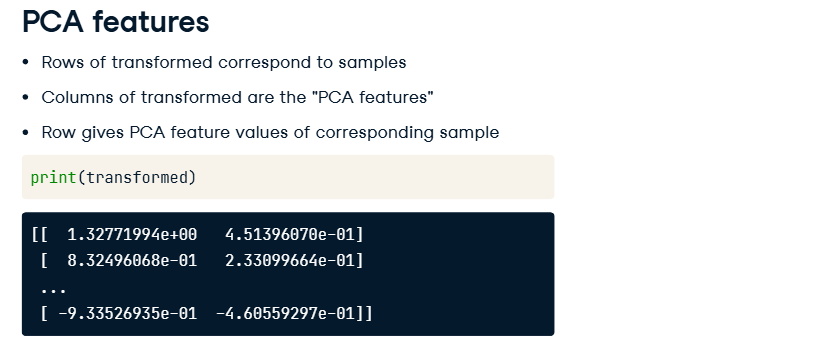
* **PCA** is **linear**: it just rotates and re-scales axes.
* **t-SNE** is **non-linear**: it focuses on preserving local neighbor relationships.
* PCA is great for **compression & preprocessing**, while t-SNE is better for **visualization**.

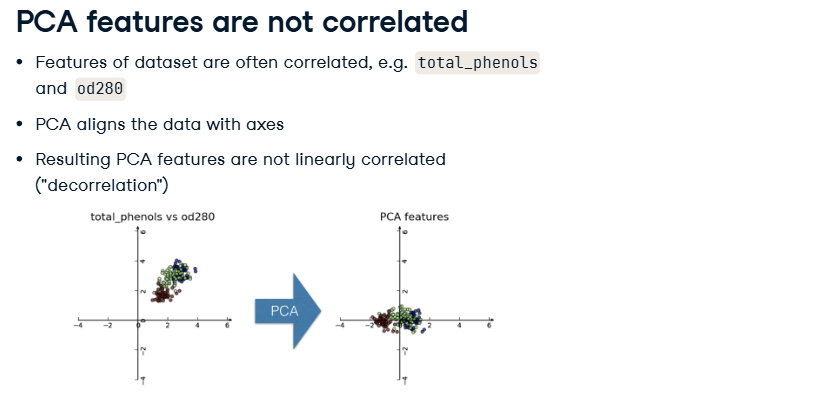
✅ **In short:**  
PCA finds the most important patterns in high-dimensional data by creating new "summary features" (principal components), allowing us to reduce dimensions while keeping most of the useful information.

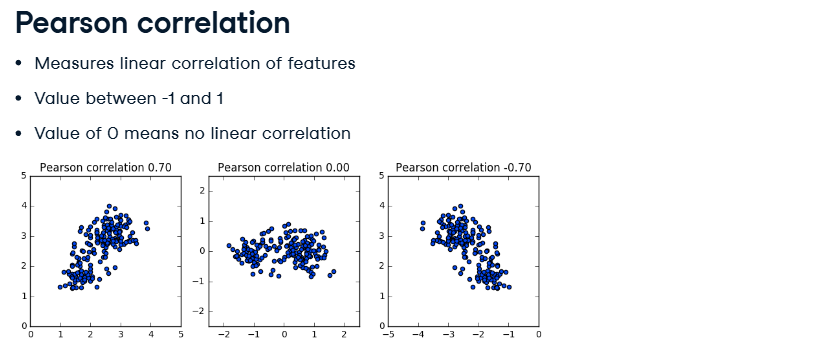


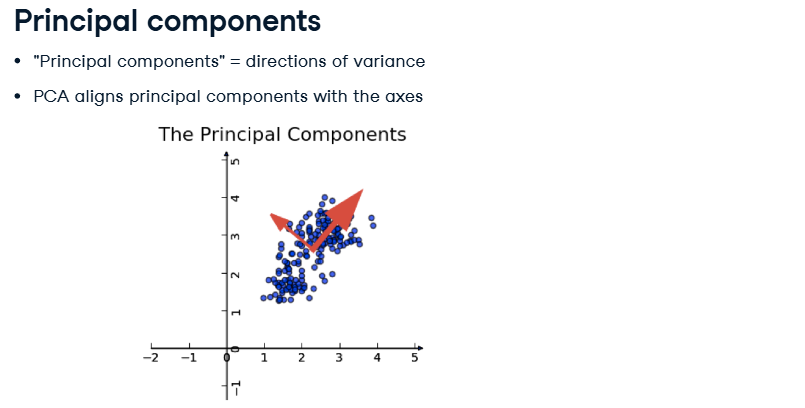


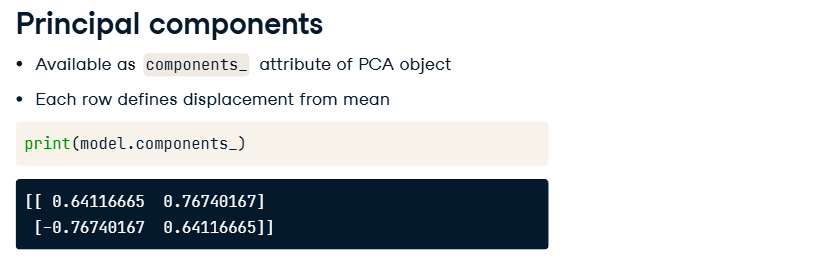


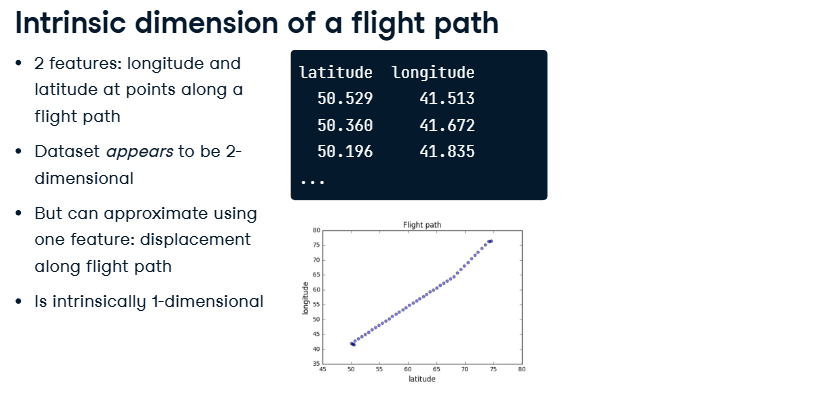


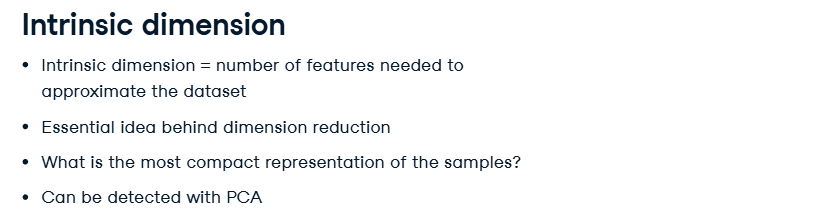


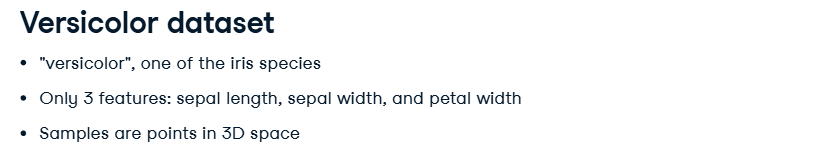


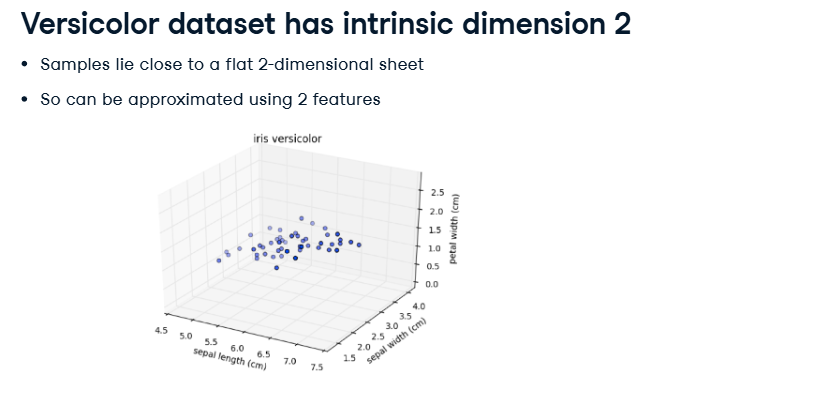


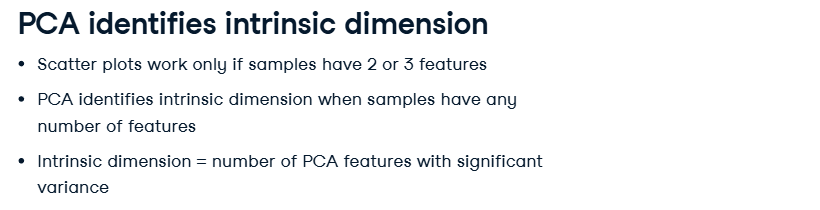


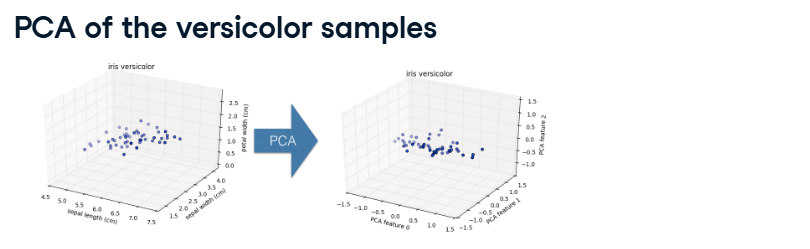


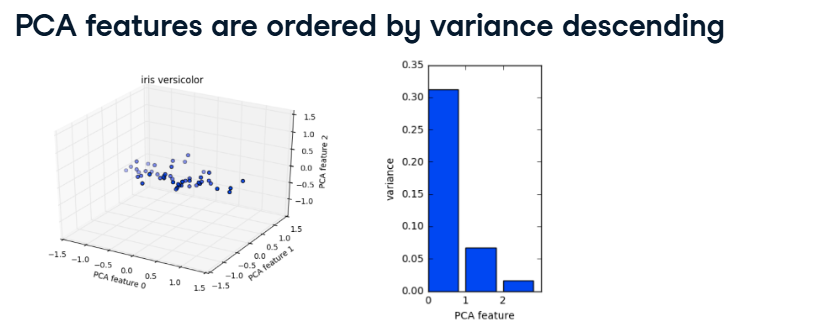


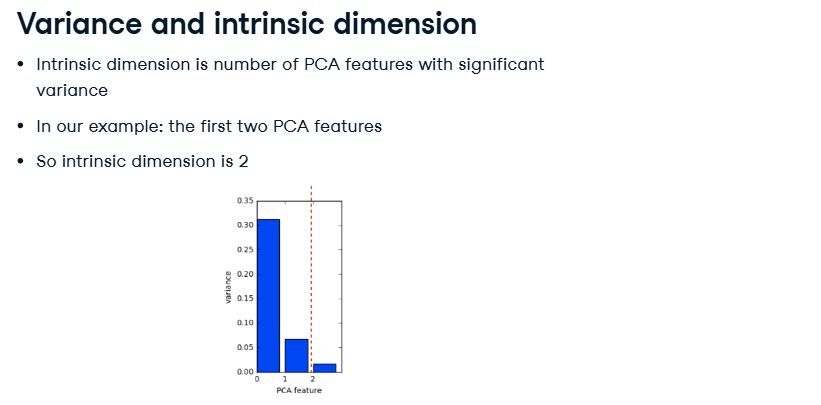


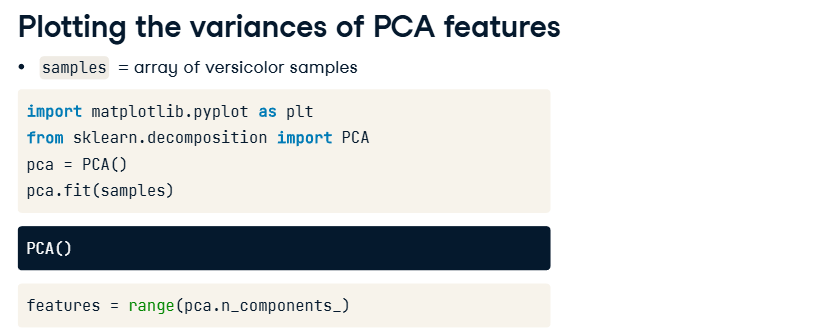


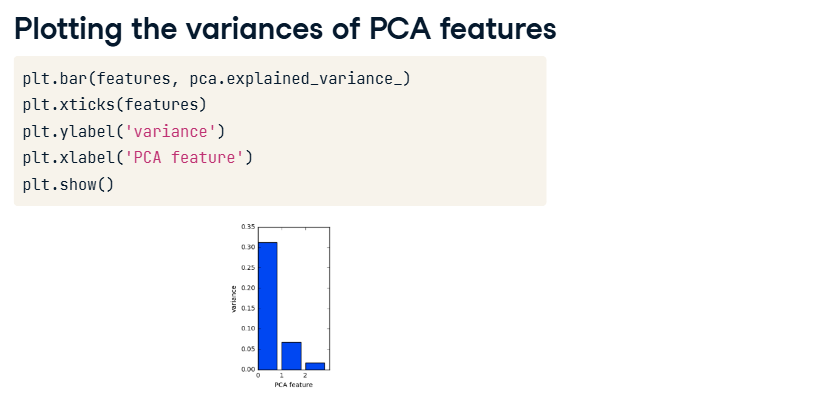


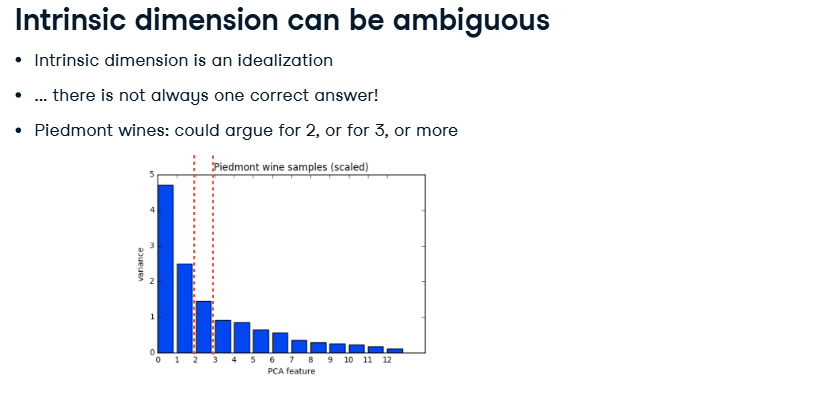


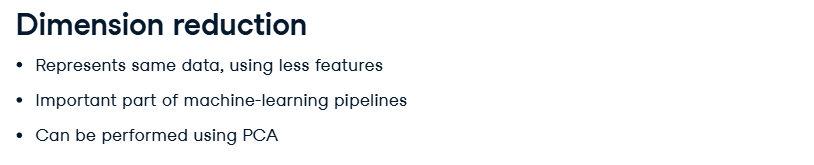


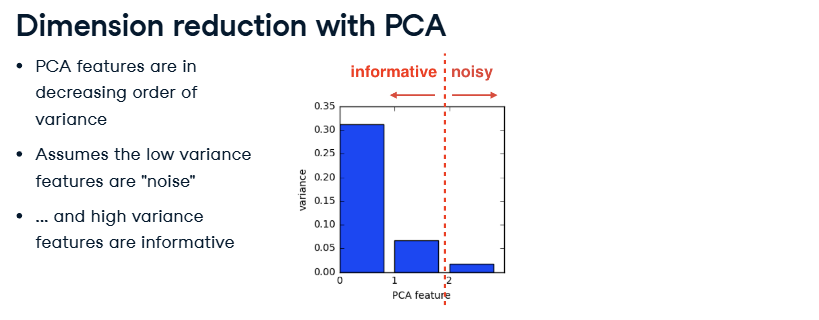


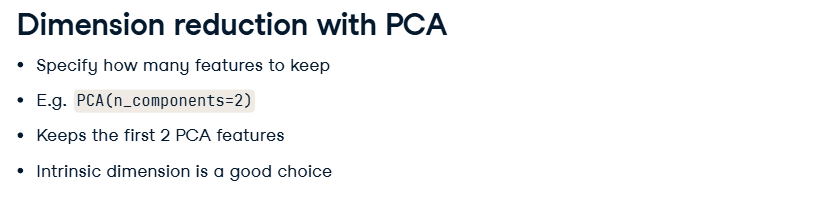


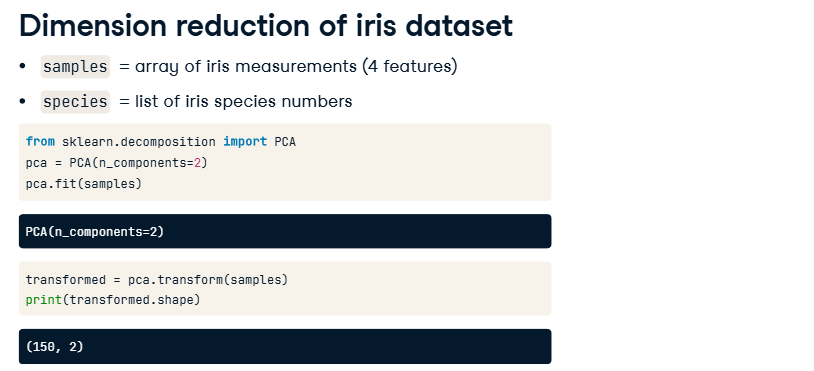


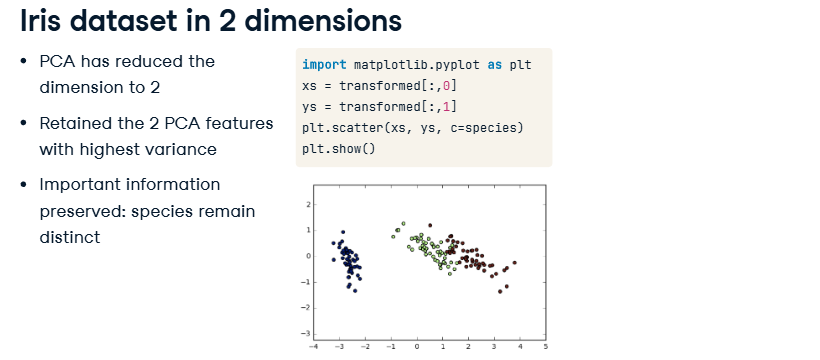


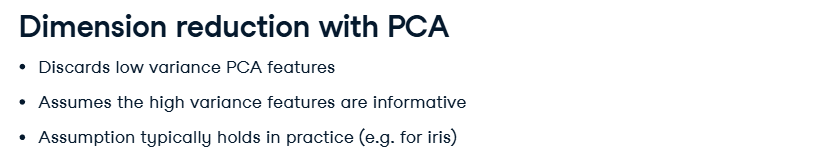


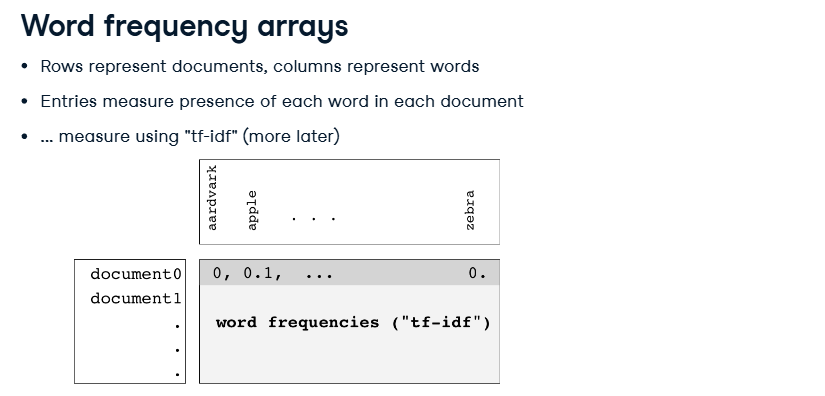


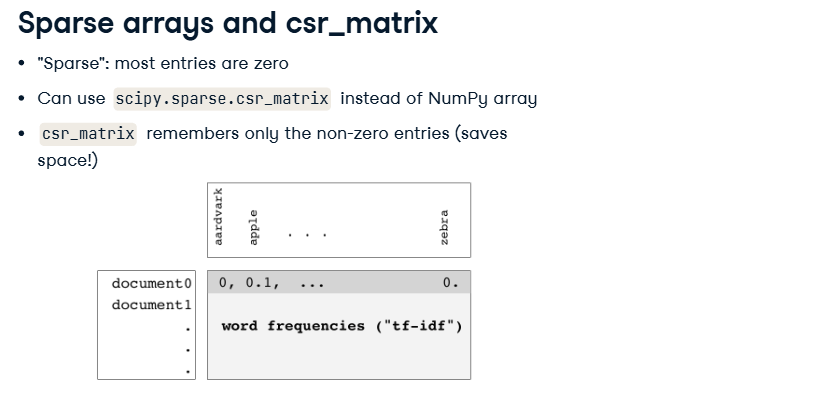






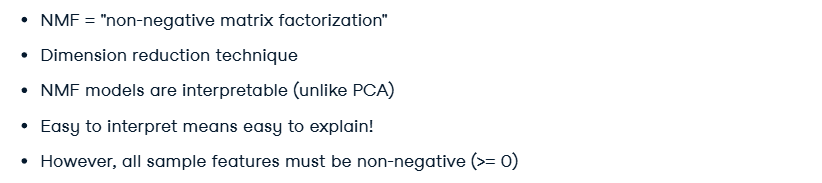


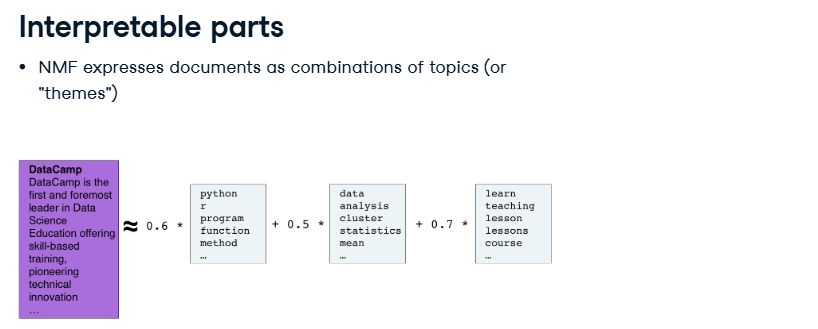


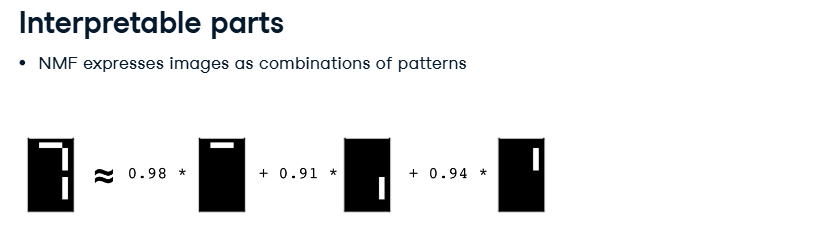


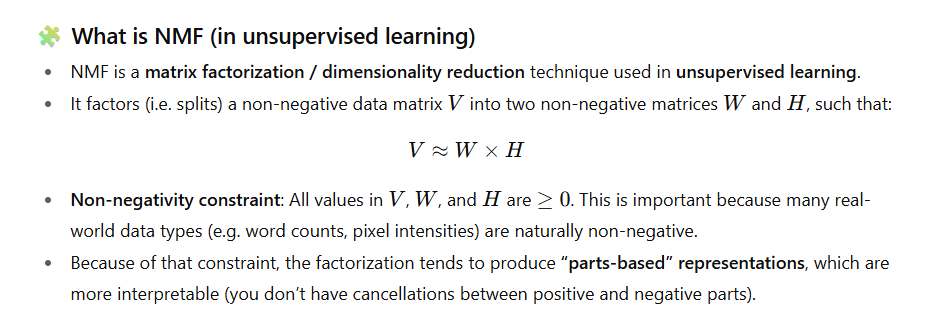


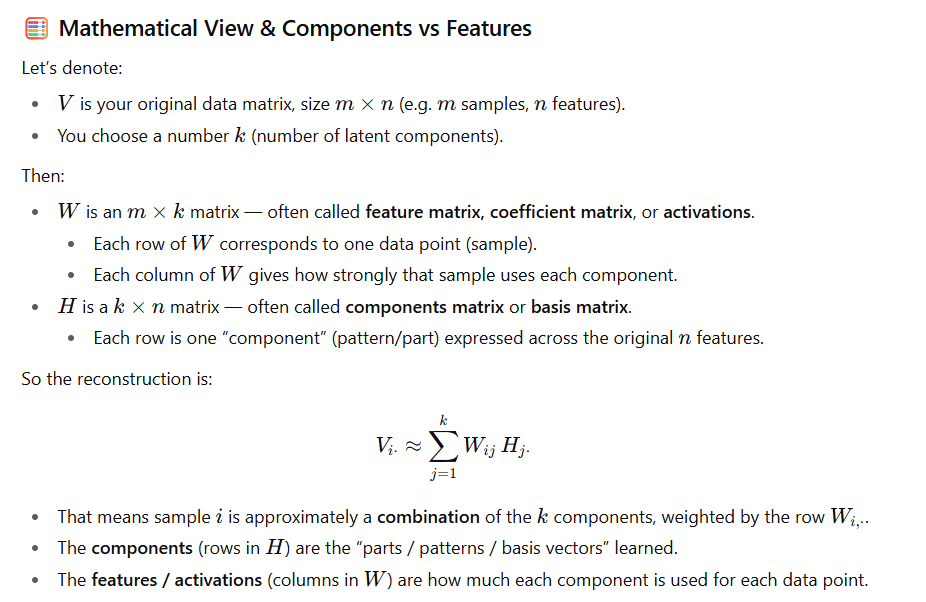
**NMF ( Non-Negative Matrix Factorization )**



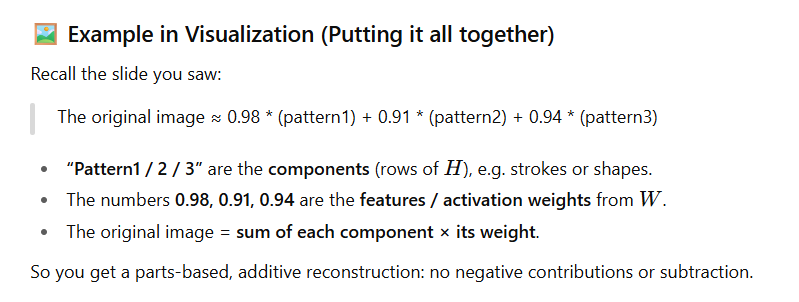




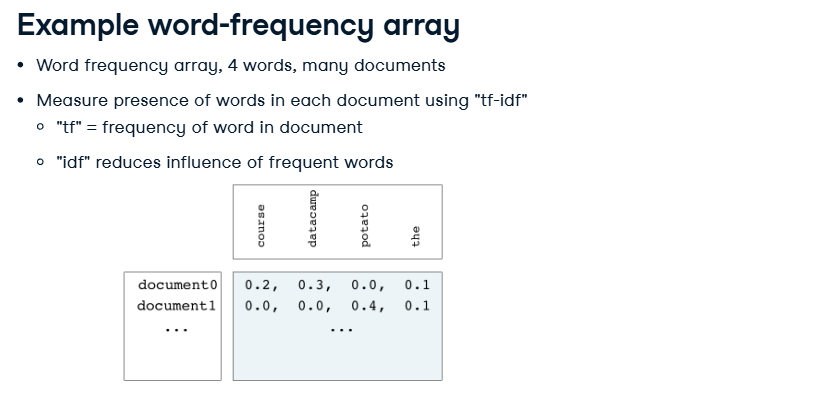


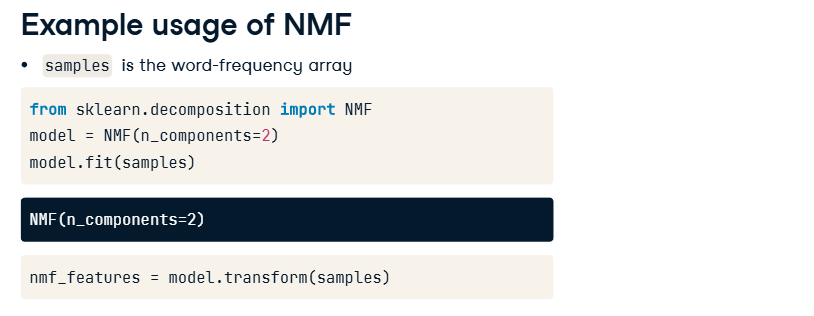


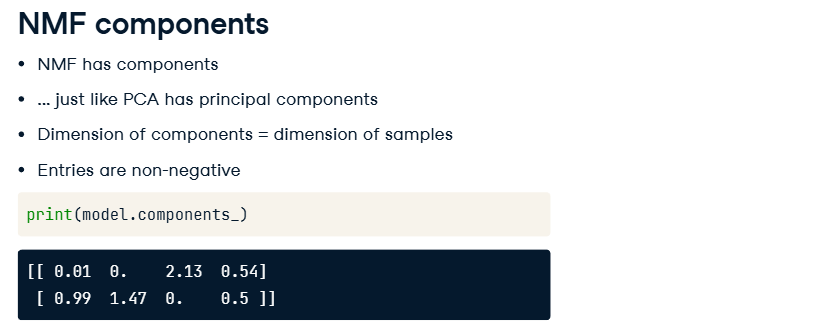


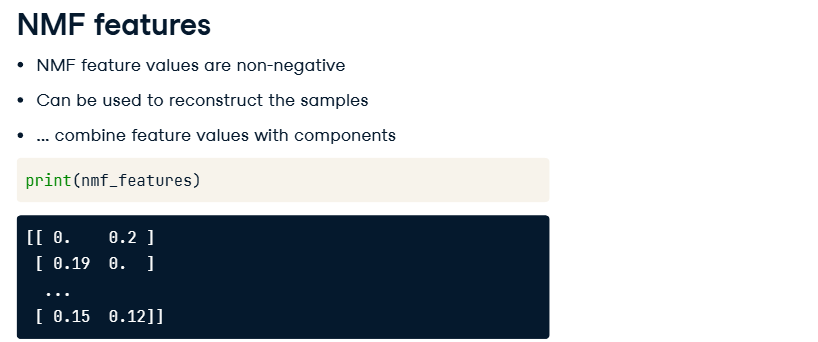


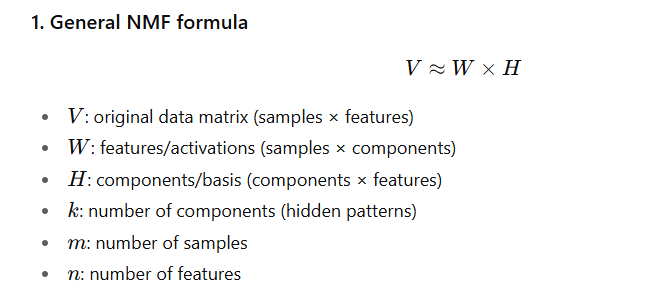


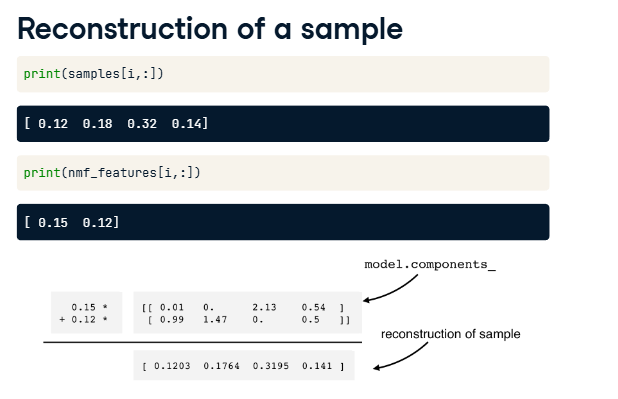


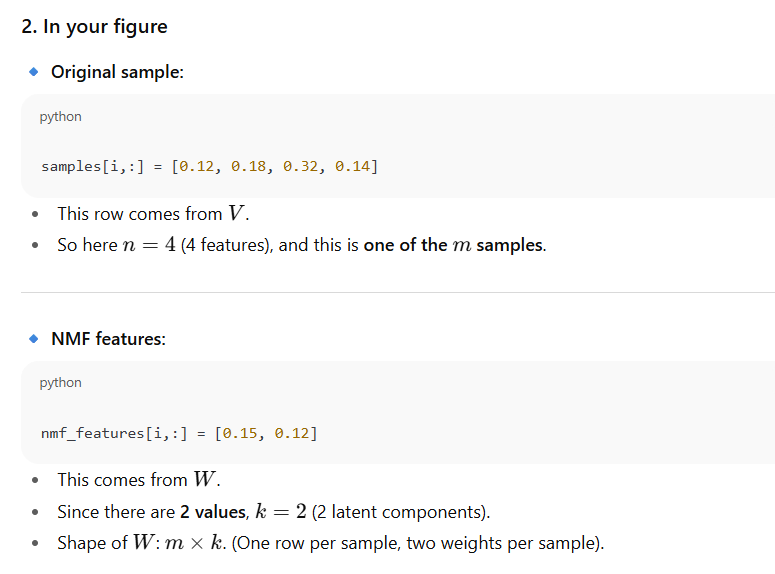


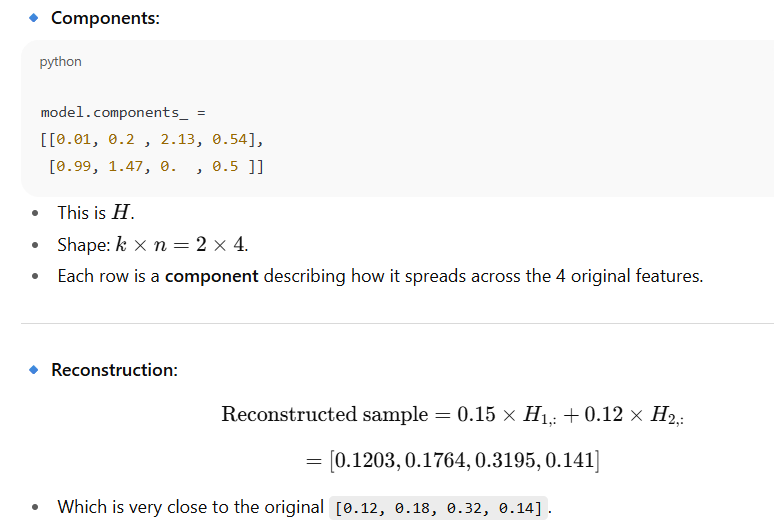


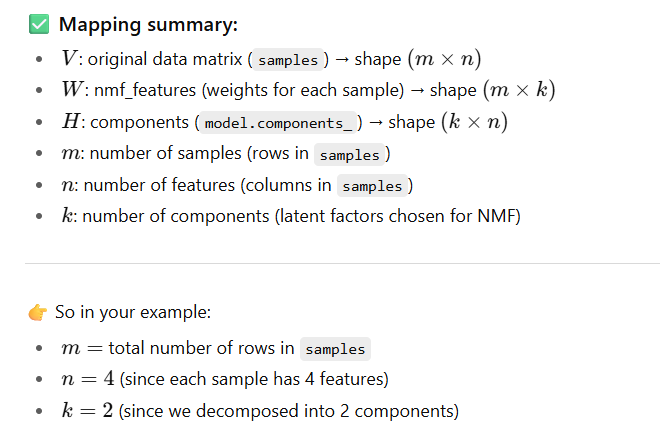


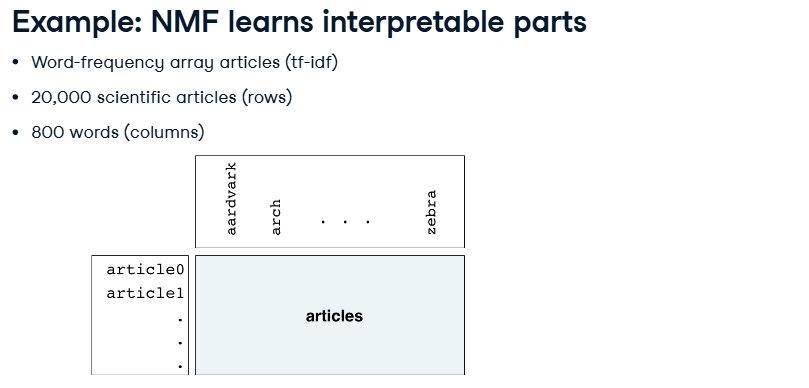




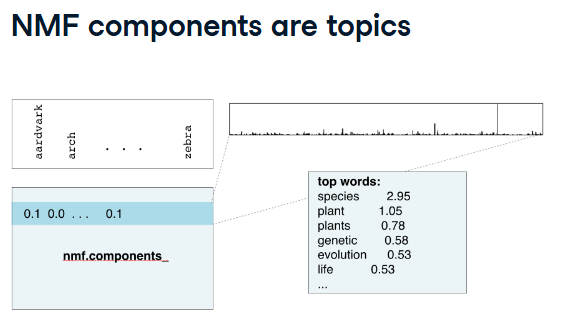




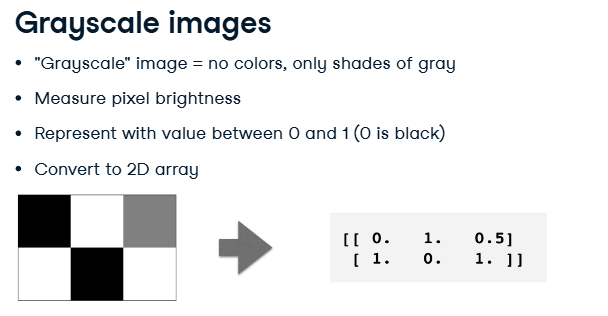


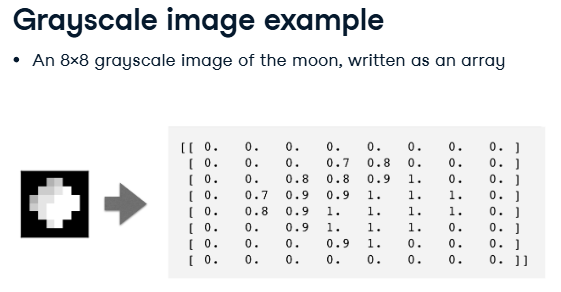


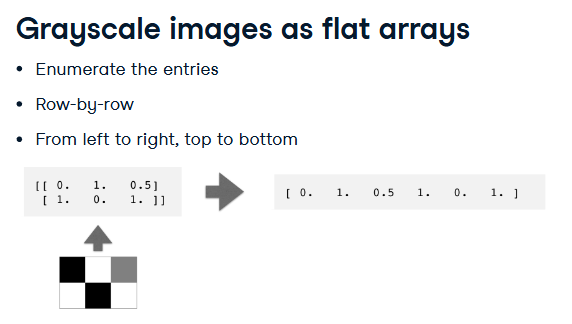


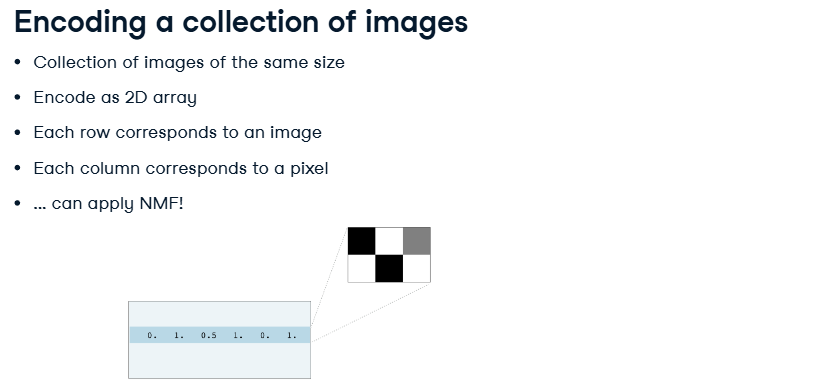


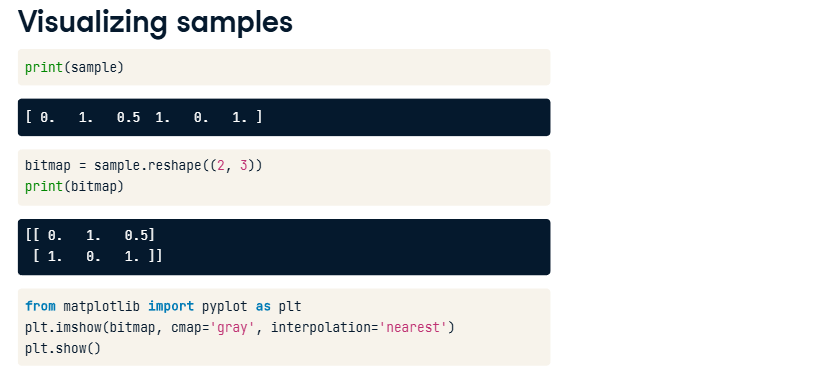












**Building Recommender Systems using NMF**

