UNIT IV

Principal Component Analysis (PCA)

What is PCA?

Principal Component Analysis (PCA) is a dimensionality reduction technique used to reduce the number of input variables in a dataset while retaining as much variability (information) as possible. It transforms the original variables into a new set of uncorrelated variables called **principal components**, ordered by the amount of variance they capture from the data.

- PCA is **unsupervised**: it doesn't use any target labels.
- PCA assumes **linearity** and that the principal components are **orthogonal**.
- It is **sensitive to scaling**; standardization is critical.

Why Use PCA?

- To simplify complex datasets with many variables.
- To remove redundant or correlated features.
- To reduce computational cost and noise.
- To visualize high-dimensional data in 2D/3D plots.

Common Terms in PCA

- **Dimensionality:** Number of features (columns) in the dataset.
- **Correlation:** Measures how strongly two variables are related (ranges from -1 to +1).
- **Orthogonal:** Variables are uncorrelated (correlation = 0).
- **Eigenvectors:** For a square matrix A, a vector v is an eigenvector if $Av=\lambda v$ (λ is a scalar).
- **Covariance Matrix:** Matrix showing covariances between variable pairs.

How PCA Works – Step-by-Step

1. **Standardize the Data**
   - Ensure all variables contribute equally by transforming them to have zero mean and unit variance.
2. **Compute the Covariance Matrix**
   - Measures how variables relate to one another (whether they vary together).
3. **Calculate Eigenvalues and Eigenvectors**
   - Eigenvectors determine the **direction** of the new feature space.
   - Eigenvalues determine the **magnitude** (importance) of the directions.
4. **Sort and Select Principal Components**
   - Rank eigenvalues from highest to lowest.
   - Select the top k eigenvectors to form a **projection matrix**.
5. **Transform the Data**
   - Multiply the standardized data with the projection matrix to get the reduced representation.

Mathematical Representation

- Let X be the original dataset with nnn samples and ppp features.
- After PCA, we obtain Z=XW, where:
  - W = matrix of top k eigenvectors (principal components)
  - Z = transformed (reduced) dataset

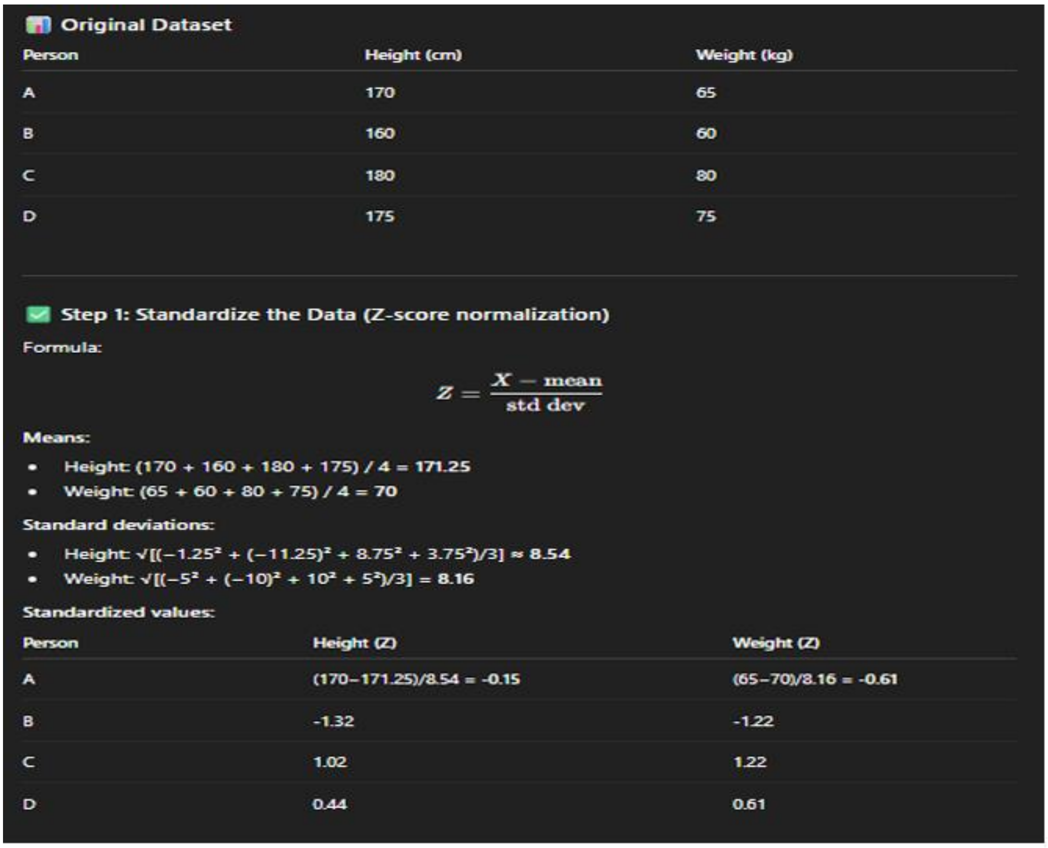
Explained Variance

- Each principal component explains a portion of the total variance.
- The **explained variance ratio** tells us how much information (variance) is retained.

- It helps in choosing the number of components to keep (e.g., enough to retain 95% variance).

Applications of PCA

- **Data Compression:** Reduces the number of features while preserving most important information.
- **Noise Reduction:** Discards components with low variance to eliminate noise.
- **Feature Extraction:** Generates new, uncorrelated features that capture the most variance.
- **Visualization of High-Dimensional Data:** Projects data into 2D or 3D space for easier visualization.
- **Preprocessing for Machine Learning:** Simplifies data, reduces dimensionality, and improves model performance and training speed.



📊 **Original Dataset**

Person	Height (cm)	Weight (kg)
A	170	65
B	160	60
C	180	80
D	175	75

✅ **Step 1: Standardize the Data (Z-score normalization)**

Formula:

$$Z = \frac{X - \text{mean}}{\text{std dev}}$$

Means:

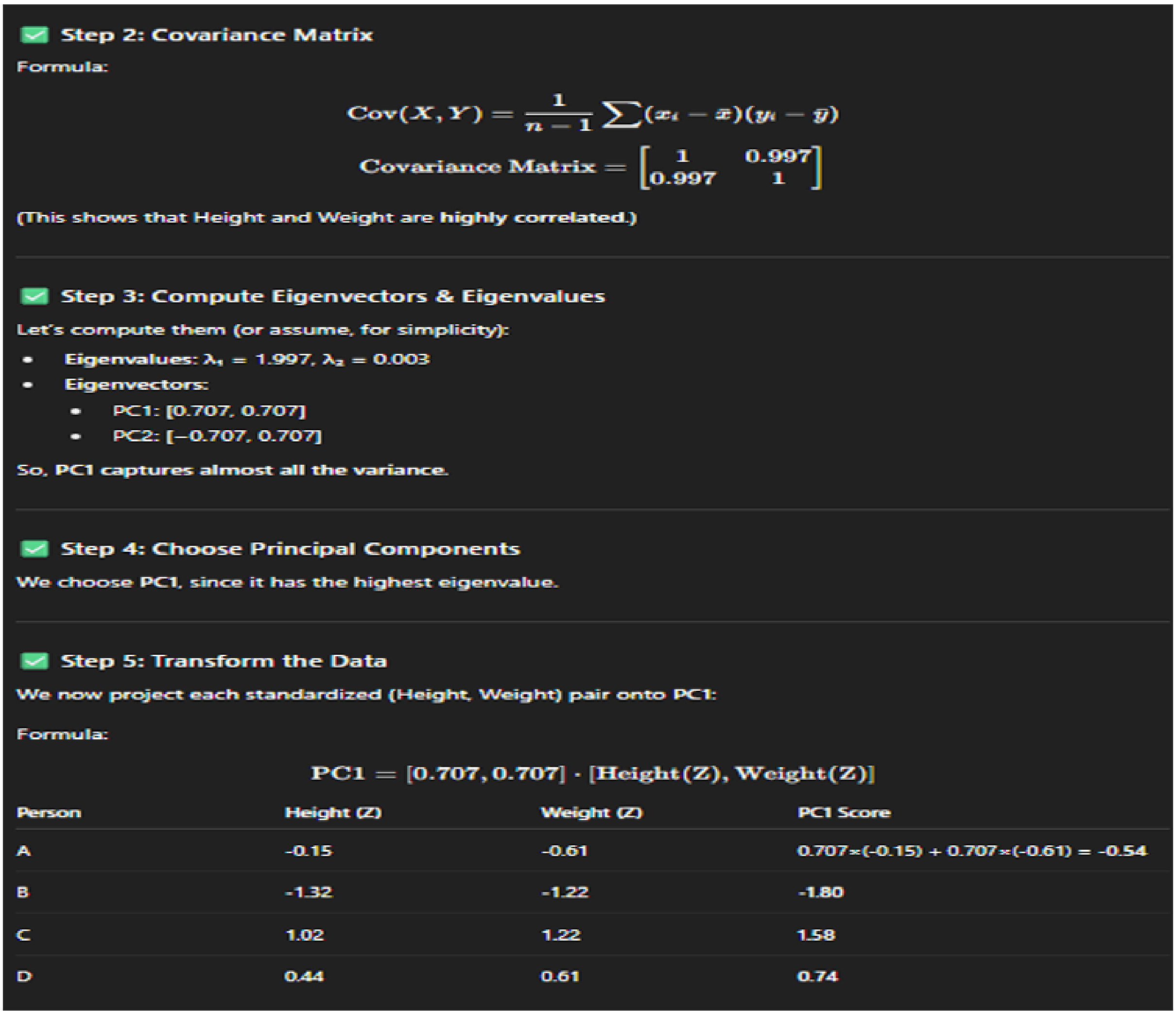
- Height: (170 + 160 + 180 + 175) / 4 = 171.25
- Weight: (65 + 60 + 80 + 75) / 4 = 70

Standard deviations:

- Height: $\sqrt{[(-1.25^2 + (-11.25)^2 + 8.75^2 + 3.75^2)/3]} \approx 8.54$
- Weight: $\sqrt{[(-5^2 + (-10)^2 + 10^2 + 5^2)/3]} = 8.16$

Standardized values:

Person	Height (Z)	Weight (Z)
A	(170−171.25)/8.54 = -0.15	(65−70)/8.16 = -0.61
B	-1.32	-1.22
C	1.02	1.22
D	0.44	0.61



✅ Step 2: Covariance Matrix

Formula:

$$\mathrm{Cov}(X, Y) = \frac{1}{n-1}\sum(x_i - \bar{x})(y_i - \bar{y})$$

$$\text{Covariance Matrix} = \begin{bmatrix} 1 & 0.997 \\ 0.997 & 1 \end{bmatrix}$$

(This shows that Height and Weight are **highly correlated**.)

✅ Step 3: Compute Eigenvectors & Eigenvalues

Let's compute them (or assume, for simplicity):

- **Eigenvalues**: $\lambda_1 = 1.997$, $\lambda_2 = 0.003$
- **Eigenvectors**:
  - PC1: $[0.707, 0.707]$
  - PC2: $[-0.707, 0.707]$

So, **PC1 captures almost all the variance**.

✅ Step 4: Choose Principal Components

We choose **PC1**, since it has the highest eigenvalue.

✅ Step 5: Transform the Data

We now project each standardized (Height, Weight) pair onto PC1:

Formula:

$$\mathrm{PC1} = [0.707, 0.707] \cdot [\mathrm{Height(Z)}, \mathrm{Weight(Z)}]$$

Person	Height (Z)	Weight (Z)	PC1 Score
A	-0.15	-0.61	0.707×(-0.15) + 0.707×(-0.61) = -0.54
B	-1.32	-1.22	-1.80
C	1.02	1.22	1.58
D	0.44	0.61	0.74

Machine Learning and its Types

Machine learning is a field of artificial intelligence that automates analytical model building.

It is based on the concept that systems can learn from data, identify patterns, and make decisions with minimal human intervention.

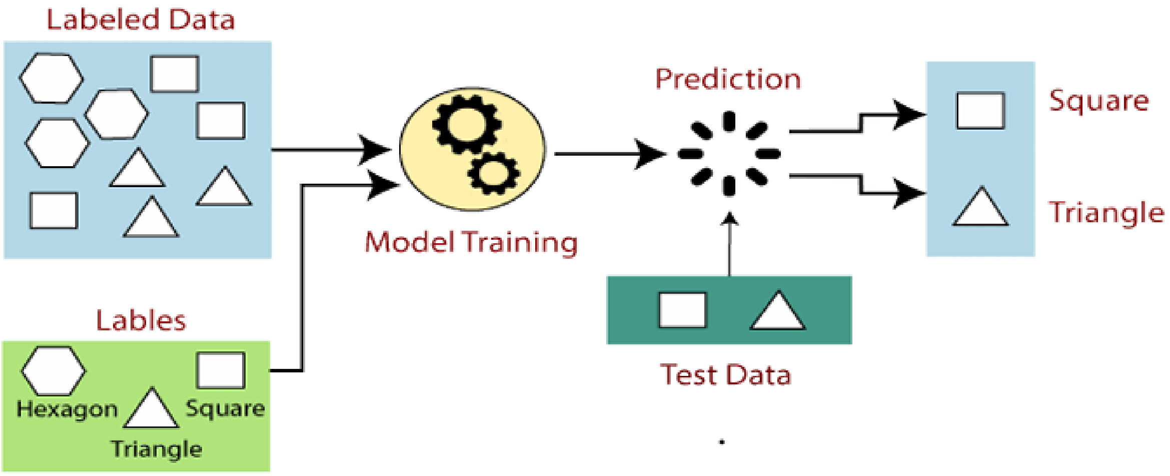
The process involves training algorithms on datasets to achieve an expected outcome, such as identifying a pattern or recognizing an object.

ML is closely related to data science, as it uses algorithms and techniques to automate data analysis and apply those insights to tasks.

Types of Machine Learning

1. Supervised Learning

In supervised learning, the algorithm is trained on a **labeled dataset**. This means the training data includes both input examples and their correct output labels. The algorithm learns the relationship between the inputs and outputs to make predictions on new, unseen data.



Examples: Fraud detection, image classification, spam filtering.

Classification is used for predicting categorical, discrete outputs. Popular algorithms include:

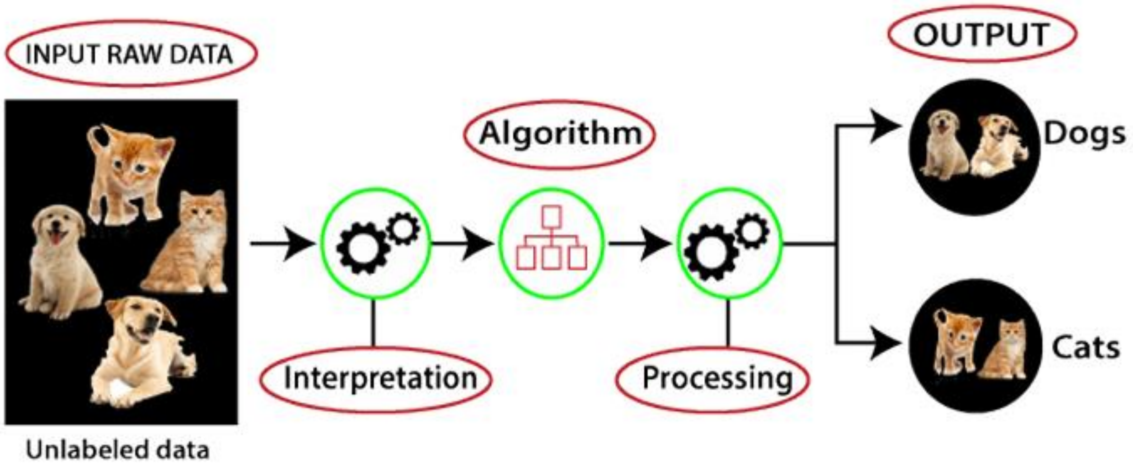
- **Decision Trees** are tree-like structures that model decisions and their consequences.
- **Random Forests** combine multiple decision trees to improve accuracy and stability.
- **Support Vector Machines (SVM)** are used to find a hyperplane that best separates data into different classes.
- **Naïve Bayes** is a probabilistic classifier based on Bayes' theorem, assuming that features are independent of each other.

Regression is used to predict continuous numeric values, such as price or age. Common techniques include:

- **Linear Regression** models the relationship between a dependent and an independent variable as a straight line.
- **Logistic Regression** is used for binary classification problems but is considered a regression method.

2. Unsupervised Learning

Unsupervised learning uses **Unlabeled data**. Instead of being given correct answers, the algorithm is left to discover hidden structures and patterns on its own.



Examples: Principal Component Analysis (PCA) and clustering.

Clustering is the process of grouping data points into clusters based on their similarities. Popular algorithms include:

- **K-Means** partitions data into a pre-defined number of clusters (K) by finding centroids and assigning data points to the nearest one.
- **Hierarchical Clustering** creates a tree-like hierarchy of clusters by either merging smaller clusters (agglomerative) or splitting a large one (divisive).
- **Gaussian Mixture Models (GMMs)** are probabilistic models used for "soft" clustering, where data points are assigned to a cluster based on the probability of belonging to a given distribution.

Dimensionality Reduction simplifies datasets by reducing the number of features while retaining the most important information.

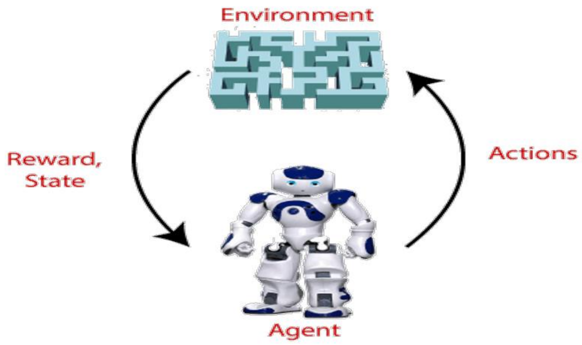
- **Principal Component Analysis (PCA)** is a statistical process that uses orthogonal transformations to convert correlated features into a set of linearly uncorrelated features called principal components.

Association Rule Mining is a rule-based technique for finding relationships between variables in a large dataset. A common application is market basket analysis, which identifies items frequently purchased together.

- The **Apriori algorithm** is a widely used technique for this purpose.

3. Reinforcement Learning

Reinforcement learning trains an agent to make decisions by interacting with an environment. The agent learns through a trial-and-error method, receiving **rewards** for favorable actions and **penalties** for unfavorable ones.



- **Examples**: Training a robot to walk or a program to play a game like Go.

Types of Reinforcement learning

There are mainly two types of reinforcement learning, which are:

o Positive Reinforcement : The positive reinforcement learning means adding something to increase the tendency that expected behaviour would occur again.

o Negative Reinforcement : The negative reinforcement learning is opposite to the positive reinforcement as it increases the tendency that the specific behaviour will occur again by avoiding the negative condition.

MACHINE LEARNING TEST AND VALIDATION :

{ in case if they ask this write intro from before and add this }

Training, Validation, and Test Sets

To properly evaluate a model, the dataset is typically split into three parts:

- **Training Set**: A subset of the data used to build and train the model. The model learns the relationships and patterns from this data.
- **Validation Set**: A subset used during the training phase to assess the model's performance and fine-tune its parameters. This helps to prevent overfitting by checking how the model performs on data it hasn't seen before, without using the final test set.
- **Test Set**: This is an independent, unseen subset of the data used for the final evaluation of the model after training is complete. It provides an unbiased estimate of the model's performance on real-world data.

Model Evaluation Metrics

- **Accuracy:** This is the most common metric for classification problems. It is the ratio of correct predictions to the total number of predictions made. It can be calculated as

(True Positive + True Negative) / Total Sample.

- **Precision:** This measures the fraction of true positives among all positive predictions made by the model. It is calculated as

True Positives / (True Positives + False Positives).

- **Recall:** This measures the fraction of actual positive predictions that were correctly identified by the model. It is calculated as

True Positives / (True Positives + False Negatives).

- **F1-score:** This metric incorporates both precision and recall to provide a single score for a model's performance. It ranges from 0 to 1, where a higher score indicates better model performance.

- **Overfitting:** This occurs when a model learns the training data too well, capturing noise and random fluctuations rather than the underlying pattern. An overfitted model performs very well on the training data but poorly on new, unseen data.
- **Underfitting:** This happens when a model is too simple to capture the underlying patterns in the training data. An underfitted model performs poorly on both the training and test data.

Decision Tree – ID3 Algorithm

The **ID3 algorithm (Iterative Dichotomiser 3)** is one of the earliest and most widely used algorithms for generating **decision trees** in machine learning.

It is mainly used for **classification problems**. The algorithm builds the decision tree by selecting the attribute that gives the **highest information gain** at each step.

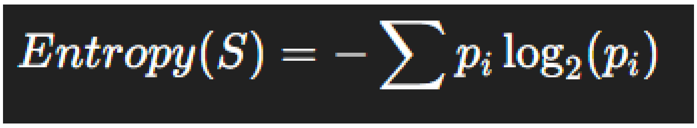
Working of ID3

1. The algorithm first calculates the **entropy** of the dataset, which measures the impurity or uncertainty in the classification.
2. For each attribute, the algorithm calculates the **information gain**. Information gain tells us how much the uncertainty is reduced if we split the dataset using that attribute.
3. The attribute with the **highest information gain** is selected as the **root node** of the decision tree.
4. The dataset is then split into subsets based on the values of the chosen attribute.
5. For each subset, the algorithm is applied again to choose the best attribute, and the process continues **recursively** until:
    - All the records are classified (pure subsets), or
    - There are no more attributes to split on.
6. Finally, the leaves of the tree represent the decision outcomes (classes).

Key Concepts

- **Entropy**:

  Entropy is a measure of disorder or impurity in a dataset. It is calculated as:



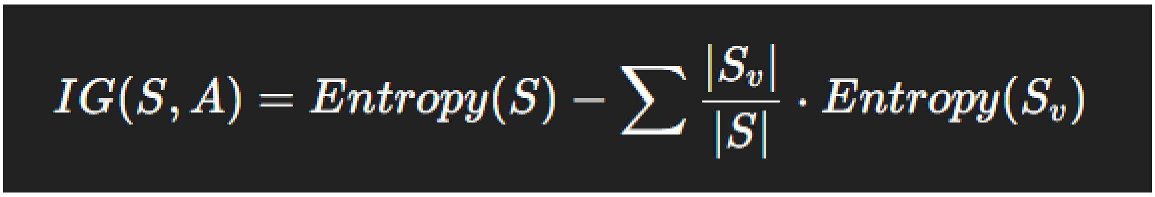
$$Entropy(S) = -\sum p_i \log_2(p_i)$$

where pip_ipi is the proportion of samples that belong to class iii.

  - Entropy = 0 means the data is perfectly classified.
  - Entropy = 1 means the data is completely impure (maximum disorder).

- **Information Gain (IG)**:

  Information gain is the reduction in entropy after splitting the dataset on an attribute. It is calculated as:



$$IG(S,A) = Entropy(S) - \sum \frac{|S_v|}{|S|} \cdot Entropy(S_v)$$

  - The attribute with the **highest information gain** is selected for the split.

Advantages of ID3

- It is easy to understand and interpret.
- It does not require normalization or scaling of data.
- It needs minimal preprocessing.

Disadvantages of ID3

- It can overfit the data if the tree becomes too large.
- The training time increases with a large number of attributes.
- The generated tree may become complex and difficult to generalize without pruning.