Feature Engineering and Optimization

Feature Management

- Feature management is the process of selecting, engineering, transforming, and handling features (input variables) in a dataset to optimize the performance of machine learning models.
- Effective feature management is crucial for improving model accuracy, interpretability, and generalization.

Feature Management

Feature Selection

Selecting the most relevant features from a dataset while eliminating redundant or irrelevant ones.

- **Methods:**
  - **Filter Methods** (e.g., Correlation, Mutual Information)
  - **Wrapper Methods** (e.g., Recursive Feature Elimination)
  - **Embedded Methods** (e.g., Lasso Regression)

Feature Engineering

Creating new features from raw data to enhance model learning.

- **Common Techniques:**
  - **Polynomial Features** (e.g., x2,x3)
  - **Domain-Specific Transformations** (e.g., Date-Time Feature Extraction)
  - **Aggregations and Grouping** (e.g., Mean Purchase Amount per User)

Feature Management

Feature Transformation
Modifying features to meet the assumptions of machine learning algorithms.
•**Techniques:**
- **Scaling** (Standardization, Min-Max Normalization)
- **Encoding** (One-Hot Encoding, Label Encoding)
- **Log Transformations** (for skewed data)

Feature Monitoring & Drift Detection
Ensuring feature stability in production over time.
•**Drift Detection Methods:**
- **Kolmogorov-Smirnov Test**
- **Population Stability Index (PSI)**
- **Chi-Square Test**

Feature Store & Feature Versioning
Managing and reusing features efficiently in ML pipelines.
•**Feature Store Tools:** Tecton, Feast, AWS SageMaker Feature Store
•**Feature Versioning:** Tracking feature changes across different ML models.

Feature selection using filter method

The **Filter Method** is a feature selection technique that evaluates the importance of features based on:

- ❑ their statistical relationships with the target variable,
- ❑ independent of the machine learning model.

This method ranks features using scoring functions and removes irrelevant or redundant ones before training a model.

Common Techniques in Filter Methods

1. Correlation Coefficient

- Measures the linear relationship between features and the target variable.
- **Pearson Correlation (r):**

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2}\sqrt{\sum(Y_i - \bar{Y})^2}}$$

  - Values close to **+1** or **-1** indicate strong relationships.
  - Features with very low correlation can be removed.

2. Chi-Square Test (For Categorical Data)

- Measures the dependency between categorical variables.
- Formula:

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

  where:

  - O = Observed frequency
  - E = Expected frequency
- Higher chi-square values indicate stronger relationships between the feature and target.

UPES
UNIVERSITY OF TOMORROW

Common Techniques in Filter Methods

Variance Threshold

Removes features with **low variance** (i.e., nearly constant across samples).

If variance of a feature is below a threshold, it is removed:

$$\text{Var}(X) = \frac{1}{n}\sum(X_i - \bar{X})^2$$

A threshold (e.g., 0.01) can be set to eliminate low-variance features.

Information Gain / Entropy (For Classification Problems)

Evaluates how well a feature splits data in classification.

Entropy:

$$H(X) = -\sum P(x)\log_2 P(x)$$

Information Gain:

$$IG = H(Y) - H(Y|X)$$

- Higher information gain means the feature contributes more to reducing uncertainty.

UPES
UNIVERSITY OF TOMORROW

Advantages of Filter Methods

Fast and scalable – Works well on high-dimensional data.

Interpretable – Uses well-known statistical measures.

Avoids overfitting – Selects features before model training.

Wrapper methods for features selection

Wrapper methods are feature selection techniques that evaluate subsets of features by **training a machine learning model** and using its performance (e.g., accuracy, F1-score) to decide which features to keep or remove.

Key Characteristics of Wrapper Methods

Model-dependent – Evaluates feature subsets by actually training a model.

Captures feature interactions – Can identify complex relationships between features.

Higher accuracy – Typically selects features that optimize model performance.

Computationally expensive – Requires training multiple models, especially on large datasets.

Types of Wrapper Methods

Forward Selection

• Starts with an **empty set** of features.

• Iteratively **adds** the best feature (based on model performance).

• Stops when adding more features does not improve performance.

Backward Elimination

• Starts with **all features** and removes the least important ones.

• At each step, the feature that has the **least impact** on model performance is removed.

• Stops when removing features **no longer improves performance**.

Recursive Feature Elimination (RFE)

•Starts with all features and recursively removes the **least important** feature.

•Uses model weights (e.g., coefficients in logistic regression) or importance scores (e.g., decision trees).

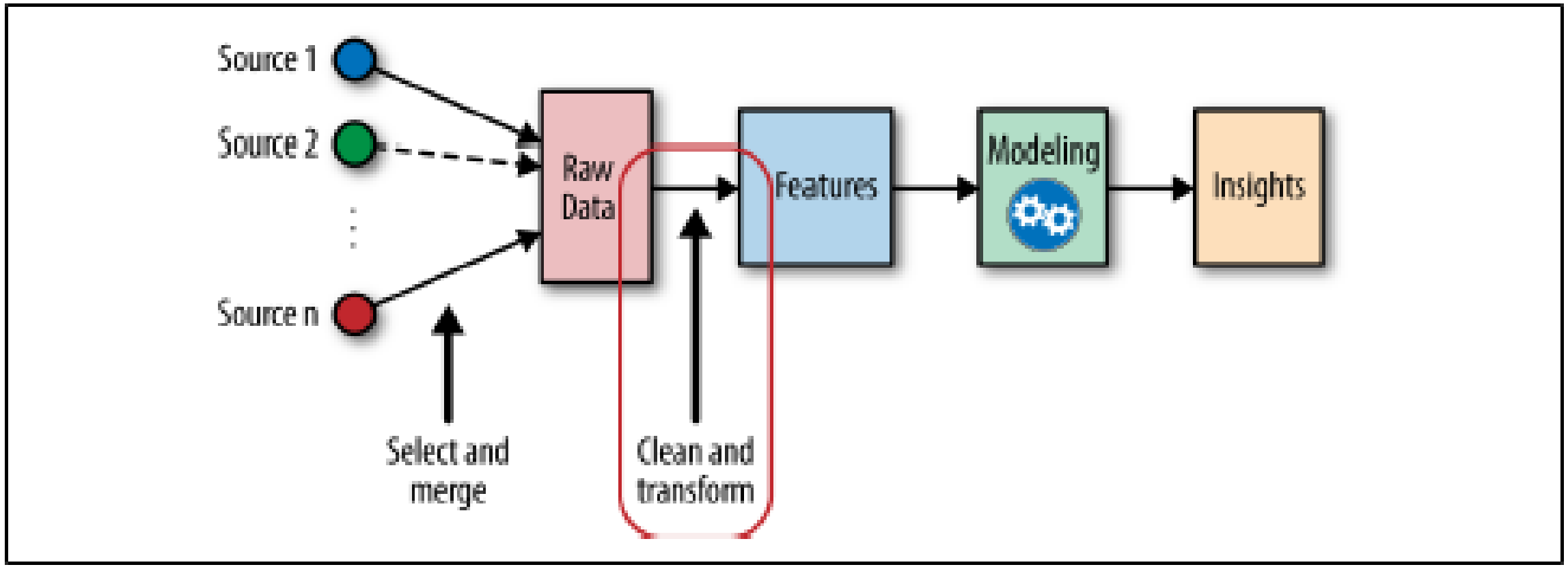
•Repeats the process until the **desired number of features** remains.

Exhaustive Feature Selection

• Evaluates **all possible subsets** of features.

• Selects the subset that gives the highest model performance.

Feature engineering in the machine learning workflow



Handling Missing Values using Statistical & Probabilistic Methods

Method 1: Mean, Median, and Mode Imputation

Mathematical Formulation

- Mean imputation:

$$X_{\text{missing}} = \frac{1}{m} \sum_{i=1}^{m} X_i$$

- Median imputation:

$$X_{\text{missing}} = \text{median}(X)$$

- Mode imputation (for categorical variables):

$$X_{\text{missing}} = \arg\max P(X)$$

Example:

Given dataset $X = \{10, 12, ?, 18, 20\}$:

$$X_{\text{mean}} = \frac{10 + 12 + 18 + 20}{4} = 15$$

Thus, $X_3 = 15$.

UPES
UNIVERSITY OF TOMORROW

k-Nearest Neighbors (k-NN) Imputation

- k-Nearest Neighbors (k-NN) imputation is a technique used to fill in missing values by finding the **k most similar instances** in the dataset and using their values to infer the missing ones.

- k-NN **preserves data variability** and captures **local structures**.

Why k-NN?

- It **leverages relationships** between features.
- It **avoids bias** introduced by using mean or median imputation.
- It can **handle both numerical and categorical data**.
- It works well when missing data follows a **non-random pattern**.

Mathematical Formulation of k-NN Imputation

Given a dataset $X \in \mathbb{R}^{m \times n}$, where some values X_{ij} are missing, we estimate X_{ij} using the k-NN algorithm.

Distance Calculation

For numerical features, we use **Euclidean distance** to measure similarity:

$$d(X_i, X_j) = \sqrt{\sum_{m=1}^{n} (X_{im} - X_{jm})^2}$$

Weighted Average for Missing Value Estimation

If feature X_i has a missing value X_{ij}, we estimate it as:

$$X_{ij} = \frac{\sum_{k=1}^{K} w_k X_{kj}}{\sum_{k=1}^{K} w_k}$$

where:

- X_{kj} are the known values of feature j from the k-nearest neighbors.
- $w_k = \frac{1}{d(X_i, X_k)}$ is the weight, inversely proportional to the distance.

UPES
UNIVERSITY OF TOMORROW

Hamming Distance

- Hamming distance is a similarity measure used to **compare categorical or binary strings** by counting the number of positions where two strings differ.

- It is widely used in k-Nearest Neighbors (k-NN) imputation when dealing with categorical data.

For two categorical vectors of equal length:

$$d_H(X, Y) = \sum_{i=1}^{n} 1(X_i \neq Y_i)$$

where:

- $d_H(X, Y)$ is the **Hamming distance** between two vectors X and Y.
- $1(X_i \neq Y_i)$ is an **indicator function**, which returns **1** if $X_i \neq Y_i$, otherwise **0**.
- n is the number of elements in the vectors.

UPES
UNIVERSITY OF TOMORROW

Example

Consider two categorical records representing a **survey response**:

ID	Color	Car Type	Fuel
1	Red	Sedan	Petrol
2	Blue	SUV	Diesel

To compute $d_H(X_1, X_2)$:

$$d_H(X_1, X_2) = 1(\text{Red} \neq \text{Blue}) + 1(\text{Sedan} \neq \text{SUV}) + 1(\text{Petrol} \neq \text{Diesel})$$

$$d_H(X_1, X_2) = 1 + 1 + 1 = 3$$

Thus, the **Hamming distance** between the two records is **3**, meaning they differ in **all three attributes**.

Example: Hamming Distance in k-NN Imputation

Dataset with Missing Values

ID	Color	Car Type	Fuel
1	Red	Sedan	Petrol
2	Blue	SUV	?
3	Green	Sedan	Diesel
4	Red	Sedan	Petrol

Compute Hamming Distance for Record 2 (Missing Fuel)

$$d_H(X_2, X_1) = 1(\text{Blue} \neq \text{Red}) + 1(\text{SUV} \neq \text{Sedan}) + 1(? \neq \text{Petrol})$$
$$= 1 + 1 + ? \text{ (Ignore missing value)}$$

$$d_H(X_2, X_3) = 1(\text{Blue} \neq \text{Green}) + 1(\text{SUV} \neq \text{Sedan}) + 1(? \neq \text{Diesel})$$
$$= 1 + 1 + ?$$

$$d_H(X_2, X_4) = 1(\text{Blue} \neq \text{Red}) + 1(\text{SUV} \neq \text{Sedan}) + 1(? \neq \text{Petrol})$$
$$= 1 + 1 + ?$$

Output

Final Imputed Dataset

ID	Color	Car Type	Fuel
1	Red	Sedan	Petrol
2	Blue	SUV	**Petrol**
3	Green	Sedan	Diesel
4	Red	Sedan	Petrol

Weighted Hamming Distance

$$d_H(X, Y) = \sum_{i=1}^{n} w_i \cdot 1(X_i \neq Y_i)$$

UPES
UNIVERSITY OF TOMORROW

Student	Math Score	Science Score	English Score
A	80	70	85
B	75	?	90
C	85	78	88
D	90	72	?
E	95	76	92

Exercise

- You are given the following dataset with missing values. Compute the mean, median, and mode for the Science Score and English Score columns.
- Impute the missing values using:
  - Mean Imputation
  - Median Imputation
  - Mode Imputation
- Which method would you choose for this dataset? Justify your answer.

Suggestion

- **Best Choice:**
- Use **Median Imputation** if the data has outliers.
- Use **Mean Imputation** if the data is normally distributed.
- **Mode Imputation** can be effective for categorical data.

Exercise

- You have a dataset representing student performance.

- Compute the Euclidean distance for the student with missing Science Score using k-NN (k = 3).
- Impute the missing value using the average of the nearest neighbors' Science Score.
- Perform the same for the missing English Score.

Student	Math Score	Science Score	English Score	Final Grade
A	80	70	85	A
B	75	?	90	B
C	85	78	88	A
D	90	72	95	A
E	95	76	?	B

Exercise

- You are analyzing customer preferences using a categorical dataset.
- Compute the Hamming Distance between:

- Customer A and B
- Customer A and C
- Customer A and D

- Interpret the results and identify which customers have similar preferences.
- Explain how Hamming Distance can be used for feature engineering in categorical data

Customer	Product Type	Payment Method	Delivery Mode
A	Electronics	Credit Card	Online
B	Clothing	Cash	Offline
C	Electronics	Credit Card	Offline
D	Electronics	Cash	Online

Filling missing values using KNN and Euclidean distance.

ID	Feature 1 (X1)	Feature 2 (X2)
1	2	4
2	1	3
3	4	?
4	6	7
5	3	5

Regression Imputation

- **Regression Imputation** is a method used to estimate missing values in a dataset by leveraging the relationships between variables.

- **Regression imputation models the missing data as a function of observed data** using linear or non-linear regression.

Why?

- Preserves relationships between variables.
- Uses available data to make better estimates rather than arbitrary replacements.
- Reduces bias compared to mean or median imputation.

Works well when data is Missing at Random (MAR).

Linear Regression for Imputation

Given a dataset X with missing values, we predict the missing values using **linear regression**:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \epsilon$$

where:

- Y = Target variable (column with missing values).
- $X_1, X_2, ..., X_n$ = Predictor variables (fully observed features).
- β_0 = Intercept term.
- $\beta_1, \beta_2, ..., \beta_n$ = Regression coefficients.
- ϵ = Error term.

To find **optimal regression coefficients** β, we minimize the sum of squared errors (SSE):

$$SSE = \sum (Y_i - \hat{Y}_i)^2$$

where $\hat{Y}_i$ is the predicted value:

$$\hat{Y} = X\beta$$

Linear Regression

- **Linear Regression** is a statistical method to model the relationship between a dependent variable (target) and one or more independent variables (features). It is commonly used for prediction and finding trends.
- $y = mx + c$

Where:

  - y = Dependent Variable (Target)
  - x = Independent Variable (Feature)
  - m = Slope of the line
  - c = Intercept (value of y when x=0)

- Formula for Slope:

$$m = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

- Formula for Intercept:

$$c = \frac{\sum y - m(\sum x)}{n}$$

Feature Construction in Machine Learning

- Feature construction is the process of creating new features from raw data to improve the performance of machine learning models.

- Well-constructed features can enhance model accuracy, interpretability, and generalization.

Feature Construction Process

Step 1: Understand the Data

•Analyze domain-specific insights.

•Identify missing values and data distributions.

Step 2: Identify Useful Transformations

•Use exploratory data analysis (EDA) to find relationships between features.

Step 3: Construct New Features

•Use domain knowledge to create meaningful features.

Step 4: Evaluate Feature Importance

•Use techniques like correlation analysis, feature importance from models, and permutation importance.

Step 5: Validate Features

•Perform cross-validation to ensure new features improve model performance.

Automated Feature Construction

- **Feature Engineering Libraries**: Featuretools, AutoFeat.
- **Deep Learning Methods**: Neural networks automatically extract high-level features.
- **Genetic Algorithms**: Used to evolve optimal feature combinations.

Best Practices

- ❑ Use domain knowledge to create meaningful features.
- ❑ Avoid data leakage (features that use future information).
- ❑ Apply feature selection to remove redundant or irrelevant features.
- ❑ Test features with baseline models before adding complexity.

Dimensionality reduction

Dimensionality reduction is a technique used in machine learning and data analysis to reduce the number of input variables (features) while retaining as much useful information as possible.

It helps in improving computational efficiency, visualization, and avoiding the curse of dimensionality.

Why Dimensionality Reduction?

- Curse of Dimensionality
- Redundancy and Correlation.
- Computational Efficiency
- Visualization

Mathematical Formulation

Let's assume a dataset $X \in \mathbb{R}^{m \times n}$, where m is the number of samples and n is the number of features.

Dimensionality reduction finds a transformation function:

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^k, \quad k < n$$

such that the new transformed dataset $Z \in \mathbb{R}^{m \times k}$ retains maximum information from X.

Principal Component Analysis (PCA)

- **Standardizing the Data**: Normalizing features to ensure uniform scaling. Scale features to have mean = 0 and variance = 1.
- **Computing the Covariance Matrix**: Measuring relationships between features.
- **Calculating Eigenvalues and Eigenvectors**: Identifying principal components (directions of maximum variance).
- **Selecting Principal Components**: Choosing top eigenvectors based on eigenvalues.
- **Projecting the Data**: Transforming original data onto the new lower-dimensional space.

Principal Component Analysis (PCA) – Advantages

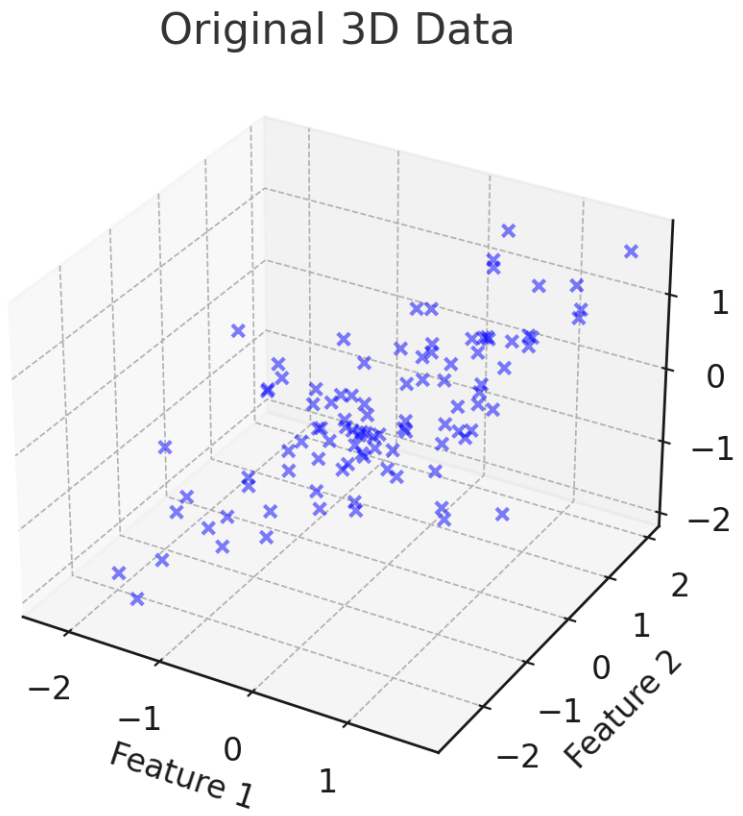
- Reduces computational complexity.
- Improves model efficiency by removing redundant features.
- Enhances detection performance by focusing on key features.
- Helps in dataset visualization by converting high-dimensional data into **2D or 3D spaces**.

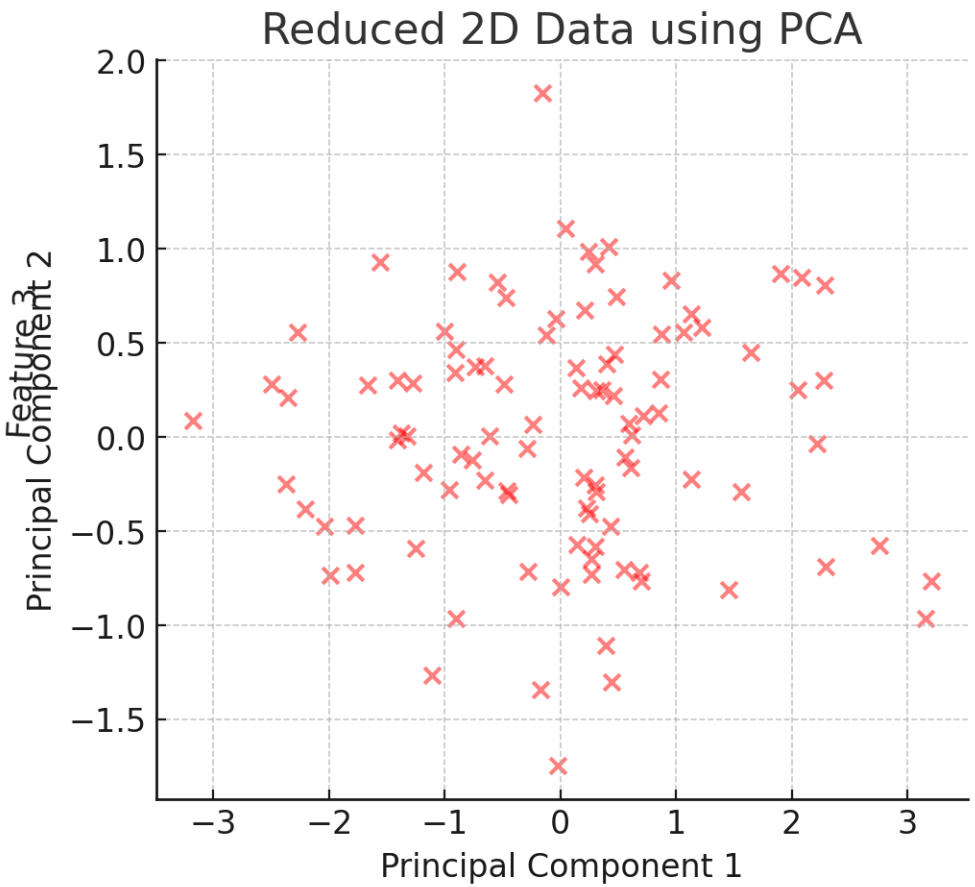
Principal Component Analysis (PCA) – limitations

- Assumes linear relationships, which may not always be present in deep learning-based forgery detection.
- Loss of information can occur if too many components are removed.
- Not effective if the variance in forged vs. original images is minimal.

Example

- Transformed from a high-dimensional space to a lower-dimensional space (e.g., 3D to 2D).





Eigenvalues & Eigenvectors

- An **eigenvector** of a matrix is a special vector that, when multiplied by the matrix, does not change its direction—only its magnitude changes.
- $Av=\lambda v$
  - A is a square matrix,
  - v is the eigenvector,
  - λ is the eigenvalue, representing the factor by which the eigenvector is scaled.
- An **eigenvalue** (λ) tells us how much the eigenvector is stretched or compressed.

Numerical example

$$A = \begin{bmatrix} 4 & 2 \\ 2 & 3 \end{bmatrix}$$

Find Eigenvalues
Eigenvalues (λ) satisfy the equation:
det(A−λI)=0

$$\begin{vmatrix} 4-\lambda & 2 \\ 2 & 3-\lambda \end{vmatrix} = 0$$

(λ−4)(λ−3)=0 So, **eigenvalues** are $\lambda_1 = 4, \lambda_2 = 3$.

Find Eigenvectors
For $\lambda_1 = 4$: Solve (A−4I)=0

$$\begin{bmatrix} 0 & 2 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

Solving, we get eigenvector: **[1, 2]**
For $\lambda_2 = 3$: Solve (A−3I)v=0(A - 3I)v = 0(A−3I)v=0

$$\begin{bmatrix} 1 & 2 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

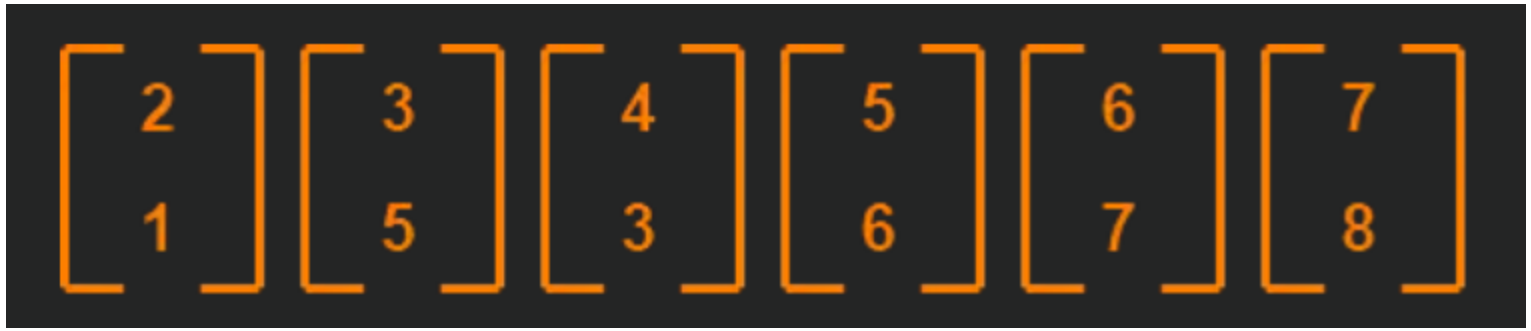
Solving, eigenvector is **[2, -1]**.

PCA Projection
- **Eigenvectors** define the new axes (Principal Components).
- **Eigenvalues** indicate the importance of each component.

Consider a dataset with 3 features and 5 samples:

Sample	Feature 1	Feature 2	Feature 3
1	2.5	2.4	3.5
2	0.5	0.7	1.2
3	2.2	2.9	3.1
4	1.9	2.2	2.8
5	3.1	3.0	4.0

- Reduce the 3 features to 2 using PCA.


2
1
3
5
4
3
5
6
6
7
7
8

Step 1: Standardize the Data

- Mean of each feature:
  - Mean of Feature 1=?
  - Mean of Feature 2=?
  - Mean of Feature 3=?
- Standard deviation of each feature:
  - Std of Feature 1=?
  - Std of Feature 2=?
  - Std of Feature 3=?

- Standardized data:

$$\text{Feature 1} = \frac{Feature1 - \mu_1}{\sigma_1}$$

$$\text{Feature 2} = \frac{Feature2 - \mu_2}{\sigma_2}$$

$$\text{Feature 3} = \frac{Feature3 - \mu_3}{\sigma_3}$$

Standardized data: ?

Step 2: Covariance Matrix

- The covariance matrix shows how the features relate to each other. The formula for the covariance between two features X and Y is:

$$\text{cov}(X, Y) = \frac{1}{n-1} \sum (X_i - \mu_X)(Y_i - \mu_Y)$$

- For our standardized data, the covariance matrix will be: ?

Step 3: Eigenvalues and Eigenvectors

Eigenvalues:

$\lambda 1 = 2.83$,

$\lambda 2 = 0.17$,

$\lambda 3 = 0.00$

Eigenvectors:

$v1 = [0.58, 0.58, 0.58]$,

$v2 = [-0.80, 0.58, 0.15]$,

$v3 = [0.16, -0.58, 0.80]$

Step 4: Select Principal Components

- We select the top 2 eigenvectors corresponding to the largest eigenvalues (since we are reducing to 2 dimensions). So, we choose v1 and v2

Step 5: Project Data

- Projected Data=Standardized Data×[v1 v2]

The reduced dataset with 2 dimensions is:

Sample	Principal Component 1	Principal Component 2
1	1.68	0.17
2	-3.12	-0.26
3	1.44	0.40
4	-0.20	-0.13
5	2.89	0.65

LDA

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction and classification technique used in machine learning and statistics.

- **Dimensionality reduction** – reducing high-dimensional data to a lower-dimensional space while maintaining class separability.
- **Classification** – improving class discrimination by projecting data onto a new axis.

LDA finds a linear combination of features that best separate two or more classes by maximizing the **between-class variance** while minimizing the **within-class variance**.

Mathematical Foundation of LDA

1. Compute the mean vectors for each class.
2. Compute the **within-class scatter matrix** (S_W) and **between-class scatter matrix** (S_B).
3. Solve the **generalized eigenvalue problem** to find the optimal projection matrix.
4. Transform the original features to a lower-dimensional space.

LDA- Example

Step 1: Consider a Small Dataset

Suppose we have two classes: **Class 1** and **Class 2**, and each class has two features (x_1, x_2).

Sample	Feature X_1	Feature X_2	Class
A	2	3	Class 1
B	3	3	Class 1
C	2	2	Class 1
D	6	7	Class 2
E	7	8	Class 2
F	6	6	Class 2

Contd…

Step 2: Compute Mean Vectors

For each class, we compute the mean feature vector (μ_1 and μ_2):

Mean vector for Class 1 (μ_1)

$$\mu_1 = \frac{1}{N_1} \sum_{i=1}^{N_1} x_i$$

$$\mu_1 = \frac{1}{3}[(2,3) + (3,3) + (2,2)]$$

$$\mu_1 = \left(\frac{2+3+2}{3}, \frac{3+3+2}{3} \right) = (2.33, 2.67)$$

Mean vector for Class 2 (μ_2)

$$\mu_2 = \frac{1}{3}[(6,7) + (7,8) + (6,6)]$$

$$\mu_2 = \left(\frac{6+7+6}{3}, \frac{7+8+6}{3} \right) = (6.33, 7)$$

Contd…

Step 3: Compute Within-Class Scatter Matrix (S_W)

The **within-class scatter matrix** measures the spread of data within each class.

$$S_W = \sum_{i=1}^{N_1} (x_i - \mu_1)(x_i - \mu_1)^T + \sum_{i=1}^{N_2} (x_i - \mu_2)(x_i - \mu_2)^T$$

For Class 1:

$$(x_A - \mu_1) = (2, 3) - (2.33, 2.67) = (-0.33, 0.33)$$

$$(x_B - \mu_1) = (3, 3) - (2.33, 2.67) = (0.67, 0.33)$$

$$(x_C - \mu_1) = (2, 2) - (2.33, 2.67) = (-0.33, -0.67)$$

Compute scatter for Class 1:

$$S_{W1} = (-0.33, 0.33)(-0.33, 0.33)^T + (0.67, 0.33)(0.67, 0.33)^T + (-0.33, -0.67)(-0.33, -0.67)^T$$

$$S_{W1} = \begin{bmatrix} (-0.33)^2 + (0.67)^2 + (-0.33)^2 & (-0.33)(0.33) + (0.67)(0.33) + (-0.33)(-0.67) \\ (-0.33)(0.33) + (0.67)(0.33) + (-0.33)(-0.67) & (0.33)^2 + (0.33)^2 + (-0.67)^2 \end{bmatrix}$$

$$= \begin{bmatrix} 0.11 + 0.45 + 0.11 & -0.11 + 0.22 + 0.22 \\ -0.11 + 0.22 + 0.22 & 0.11 + 0.11 + 0.45 \end{bmatrix}$$

$$= \begin{bmatrix} 0.67 & 0.33 \\ 0.33 & 0.67 \end{bmatrix}$$

UPES
UNIVERSITY OF TOMORROW

Contd…

for **Class 2**, computing in the same way:

$$S_{W2} = \begin{bmatrix} 0.67 & 0.67 \\ 0.67 & 1.33 \end{bmatrix}$$

$$S_W = S_{W1} + S_{W2} = \begin{bmatrix} 0.67 + 0.67 & 0.33 + 0.67 \\ 0.33 + 0.67 & 0.67 + 1.33 \end{bmatrix} = \begin{bmatrix} 1.33 & 1.00 \\ 1.00 & 2.00 \end{bmatrix}$$

Compute Between-Class Scatter Matrix (S_B)

$$S_B = N_1(\mu_1 - \mu)(\mu_1 - \mu)^T + N_2(\mu_2 - \mu)(\mu_2 - \mu)^T$$

$$\mu = \frac{\mu_1 + \mu_2}{2}$$

$$S_B = 3 \times (-2, -2.16)(-2, -2.16)^T + 3 \times (2, 2.17)(2, 2.17)^T$$

$$= 3 \times \begin{bmatrix} 4 & 4.32 \\ 4.32 & 4.66 \end{bmatrix} + 3 \times \begin{bmatrix} 4 & 4.34 \\ 4.34 & 4.71 \end{bmatrix}$$

$$= \begin{bmatrix} 12 & 12.96 \\ 12.96 & 13.98 \end{bmatrix} + \begin{bmatrix} 12 & 13.02 \\ 13.02 & 14.13 \end{bmatrix}$$

$$= \begin{bmatrix} 24 & 25.98 \\ 25.98 & 28.11 \end{bmatrix}$$

Contd…

Find Eigen values:

$$\left|S_W^{-1}S_B - \lambda I\right| = 0$$

Find Eigen vector:

$$\left(S_W^{-1}S_B - \lambda I\right)\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 0$$

OR

$$w^* = S_W^{-1}(\mu_1 - \mu_2)$$

Task: Exploring Dataset Sources and Feature Selection for Predictive Modeling