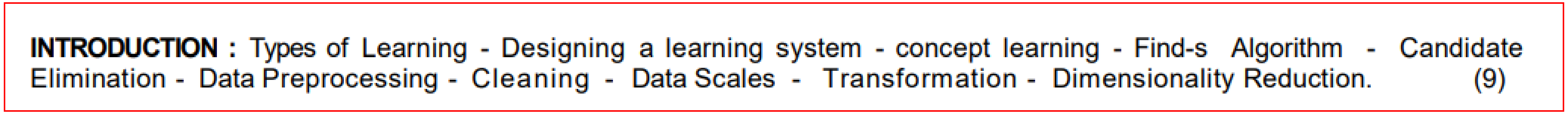
19Z601- MACHINE LEARNING

UNIT- 1 INTRODUCTION



INTRODUCTION : Types of Learning - Designing a learning system - concept learning - Find-s Algorithm - Candidate Elimination - Data Preprocessing - Cleaning - Data Scales - Transformation - Dimensionality Reduction. (9)

Presented by

Ms.Anisha.C.D

Assistant Professor

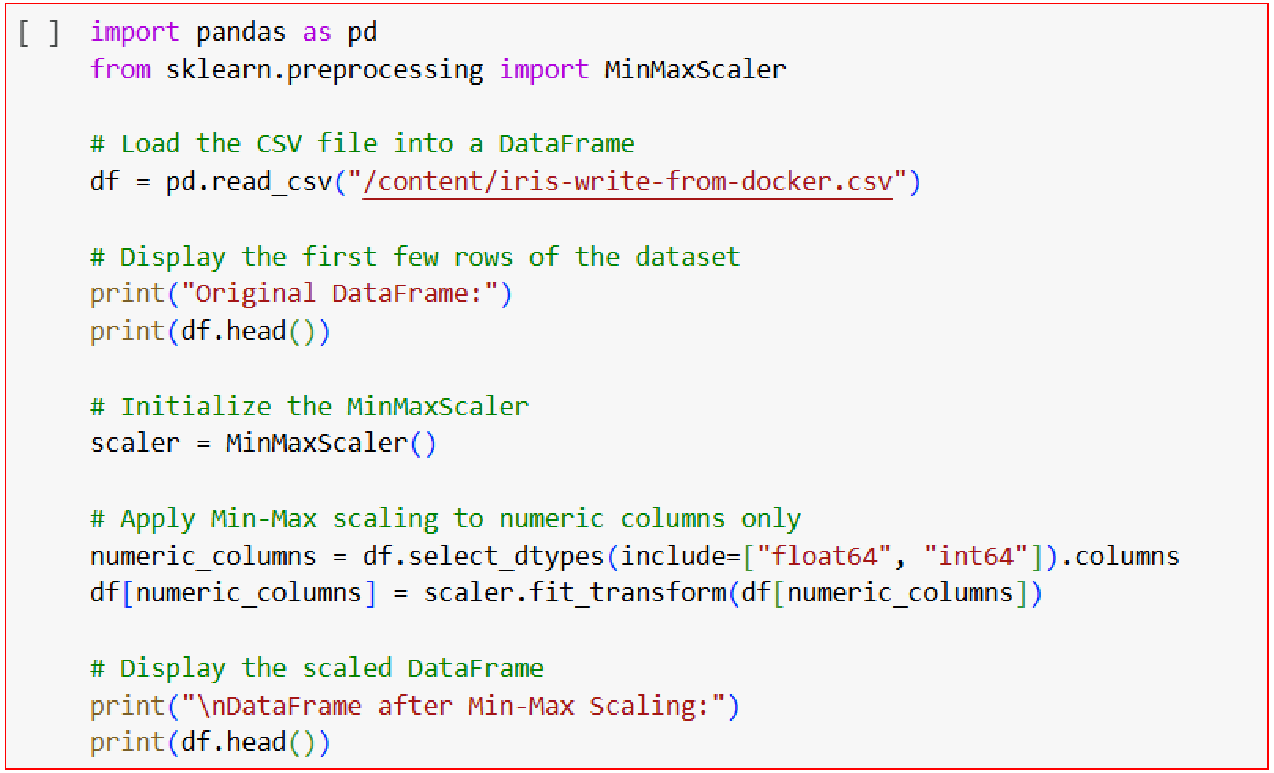
CSE

RECAP – TRANSFORMATION

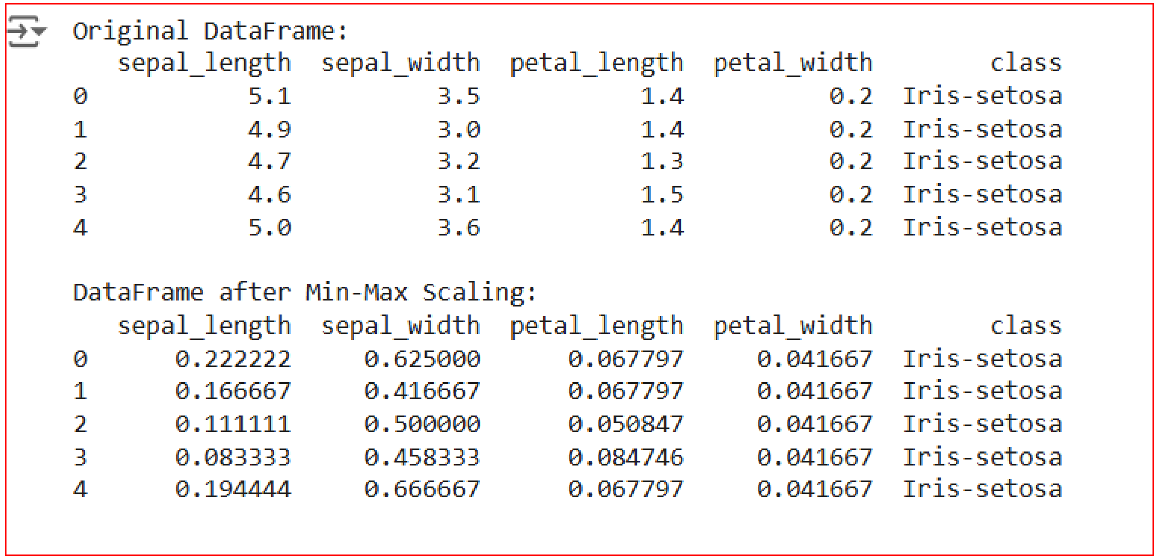
- The problem of transforming raw data into dataset is called feature engineering.

One Hot Encoding	Transformation of categorical feature into **several binary codes is called One Hot Encoding .** Example : Categorical Feature “Colors” with three possible values : Red, Yellow and Green, Transform this feature into a vector of three numerical values. Red = [1,0,0] Yellow = [0,1,0] Green = [0,0,1]
Binning	Transformation of numerical feature into categorical one. Binning is also called bucketing is the process of converting a continuous feature into multiple binary features called bines or buckets.
Normalization	Process of converting an actual range of values which a numerical feature can take into a standard range of values, **typically in the interval [-1,1] or [0,1]**
Standardization	**Z-score Normalization (Standardization)** is the procedure during which the feature values are rescaled so that they have the properties of a **standard normal distribution with mean = 0 and standard deviation =1 .**

Min Max Normalization



```
[ ] import pandas as pd
    from sklearn.preprocessing import MinMaxScaler

    # Load the CSV file into a DataFrame
    df = pd.read_csv("/content/iris-write-from-docker.csv")

    # Display the first few rows of the dataset
    print("Original DataFrame:")
    print(df.head())

    # Initialize the MinMaxScaler
    scaler = MinMaxScaler()

    # Apply Min-Max scaling to numeric columns only
    numeric_columns = df.select_dtypes(include=["float64", "int64"]).columns
    df[numeric_columns] = scaler.fit_transform(df[numeric_columns])

    # Display the scaled DataFrame
    print("\nDataFrame after Min-Max Scaling:")
    print(df.head())
```

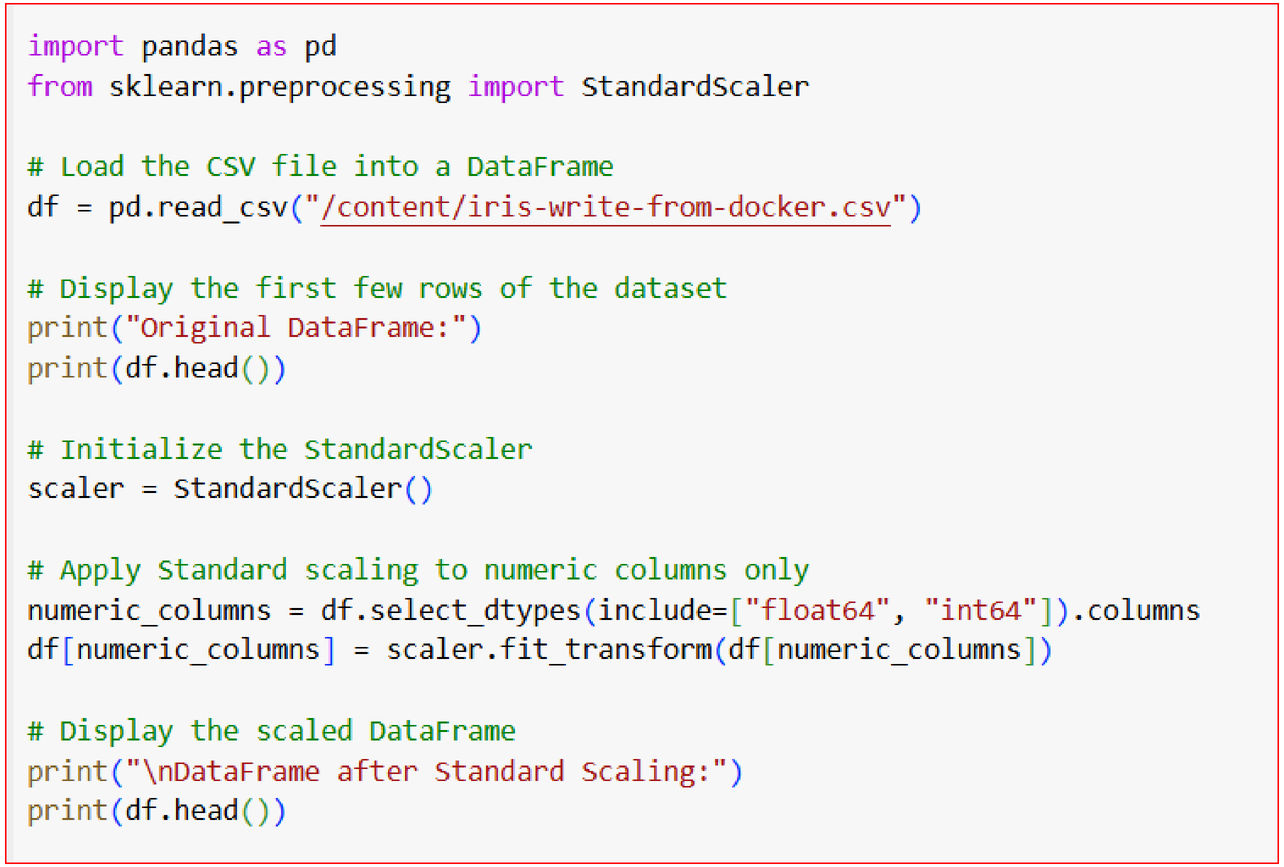


```
Original DataFrame:
   sepal_length  sepal_width  petal_length  petal_width        class
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa

DataFrame after Min-Max Scaling:
   sepal_length  sepal_width  petal_length  petal_width        class
0      0.222222     0.625000      0.067797     0.041667  Iris-setosa
1      0.166667     0.416667      0.067797     0.041667  Iris-setosa
2      0.111111     0.500000      0.050847     0.041667  Iris-setosa
3      0.083333     0.458333      0.084746     0.041667  Iris-setosa
4      0.194444     0.666667      0.067797     0.041667  Iris-setosa
```

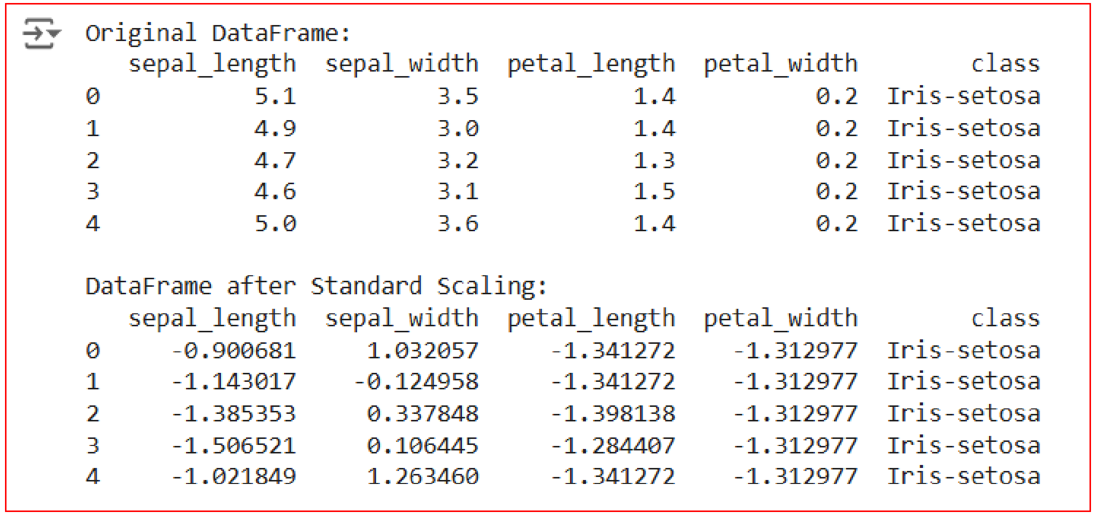
$$X_{Scaled} = \frac{X - X_{min}}{\max - min} * (\text{new max} - \text{new min}) + \text{new min}$$

What is the Mathematics behind Min Max Normalization?

Z Score Normalization (Standardization)



```python
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load the CSV file into a DataFrame
df = pd.read_csv("/content/iris-write-from-docker.csv")

# Display the first few rows of the dataset
print("Original DataFrame:")
print(df.head())

# Initialize the StandardScaler
scaler = StandardScaler()

# Apply Standard scaling to numeric columns only
numeric_columns = df.select_dtypes(include=["float64", "int64"]).columns
df[numeric_columns] = scaler.fit_transform(df[numeric_columns])

# Display the scaled DataFrame
print("\nDataFrame after Standard Scaling:")
print(df.head())
```



```
Original DataFrame:
   sepal_length  sepal_width  petal_length  petal_width        class
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa

DataFrame after Standard Scaling:
   sepal_length  sepal_width  petal_length  petal_width        class
0     -0.900681     1.032057     -1.341272    -1.312977  Iris-setosa
1     -1.143017    -0.124958     -1.341272    -1.312977  Iris-setosa
2     -1.385353     0.337848     -1.398138    -1.312977  Iris-setosa
3     -1.506521     0.106445     -1.284407    -1.312977  Iris-setosa
4     -1.021849     1.263460     -1.341272    -1.312977  Iris-setosa
```

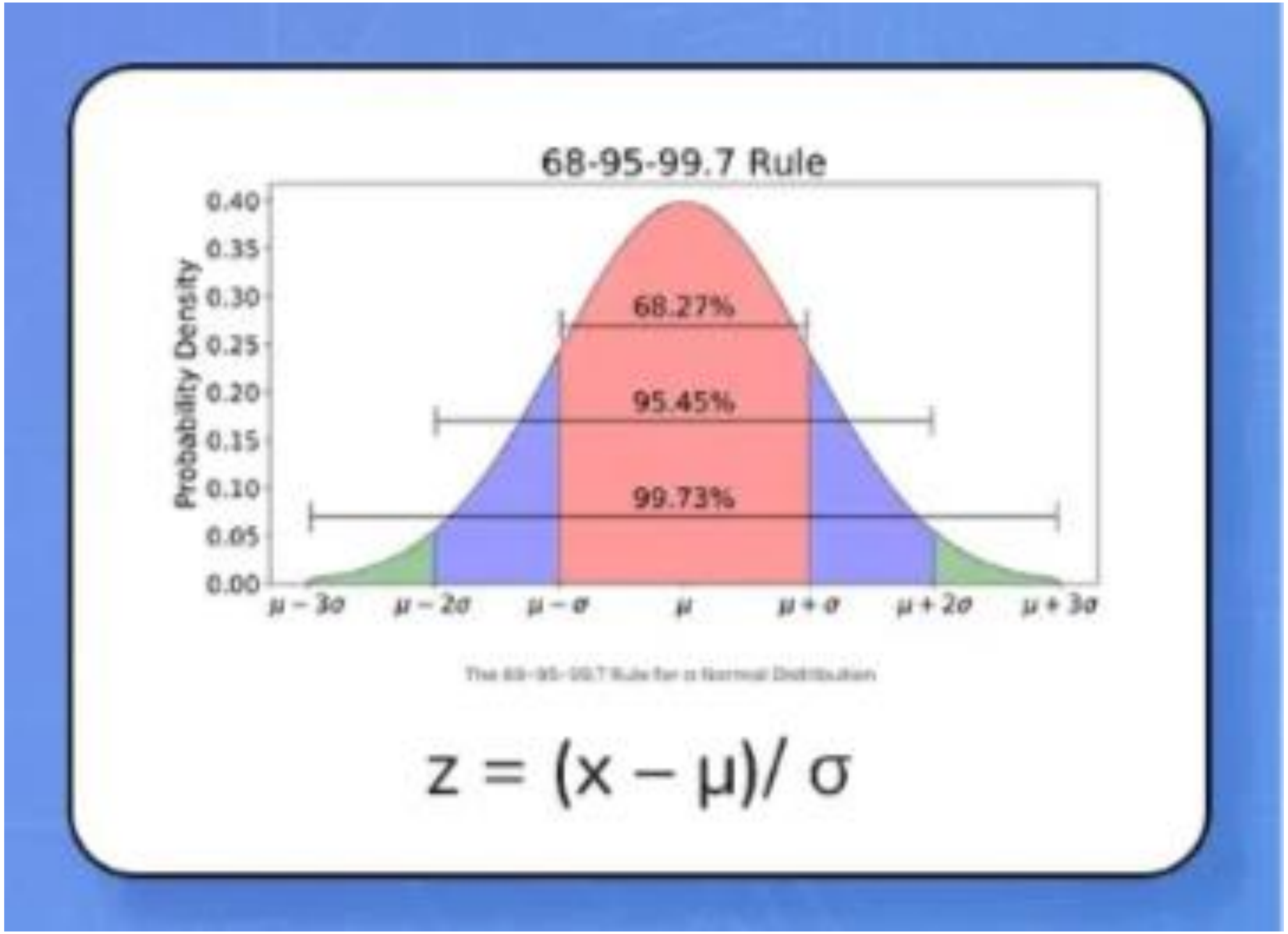
Mean

$$Z = \frac{x - \mu}{\sigma}$$

Standard Deviation

What is the Mathematics behind Standardization?

Z Score Normalization (Standardization)



The 68-95-99.7 Rule for a Normal Distribution

$$z = (x - \mu)/ \sigma$$

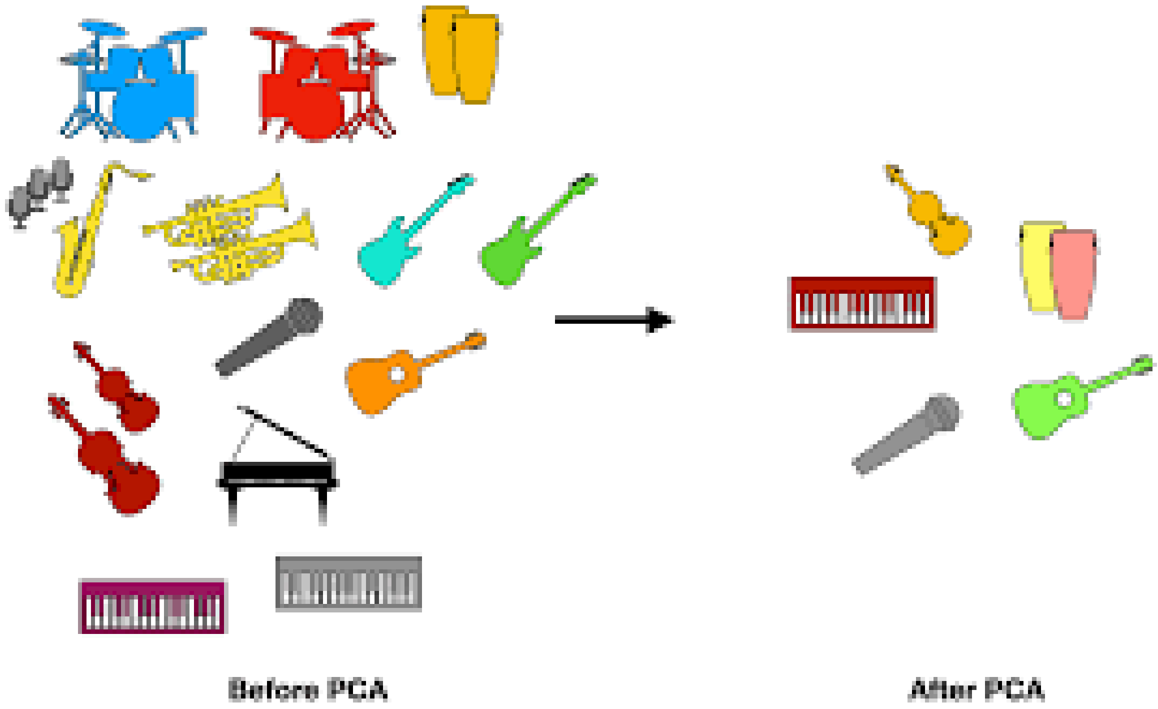
DIMENSIONALITY REDUCTION

- There are two methods for dimensionality reduction :

METHODS	DESCRIPTION
Feature Selection	Finding K of the d dimension which gives more information and discard (d-k) dimension. Example : Subset Feature Selection
Feature Extraction	Finding new set of k dimensions which is a combination of original d dimensions. **Examples :** **Linear Projection Methods :** • **Principal Component Analysis (PCA) – Unsupervised Learning** • Linear Discriminant Analysis – Supervised Learning

Principal Component Analysis (PCA)

Let’s Understand through an analogy



Aim : To reduce redundancy in dataset.

Outcome : Features -> Principal Components

Task : PCA

Form a Team of 2 Members : Create an anology for PCA Concept.

Principal Component Analysis

- **Step 1 :** Standardization
- **Step 2 :** Co-variance Matrix Computation
- **Step 3 :** Computation of Eigen vectors and Eigen Values of the Covariance matrix to identify the Principal Components.
- **Step 4 :** Create a Feature Vector
- **Step 5 :** Recast the data along the Principal Components Axes