Module 6 : Feature Extraction

Feature extraction is a process in machine learning and data analysis that involves identifying and extracting relevant features from raw data. These features are later used to create a more informative dataset, which can be further utilized for various tasks such as:

- Classification
- Prediction
- Clustering

Feature extraction aims to reduce data complexity (often known as "data dimensionality") while retaining as much relevant information as possible. This helps to improve the performance and efficiency of machine learning algorithms and simplify the analysis process. Feature extraction may involve the creation of new features ("feature engineering") and data manipulation to separate and simplify the use of meaningful features from irrelevant ones.

What is Feature Extraction?

Feature extraction is the process of identifying and selecting the most important information or characteristics from a data set. It's like distilling the essential elements, helping to simplify and highlight the key aspects while filtering out less significant details. It's a way of focusing on what truly matters in the data.

Why feature Extraction is essential?

Feature extraction is important because it makes complicated information simpler. In things like computer learning, it helps find the most crucial patterns or details, making computers better at predicting or deciding things by focusing on what matters in the data.

Common Feature Extraction Techniques

1. The need for Dimensionality Reduction

In real-world machine learning problems, there are often too many factors (features) on the basis of which the final prediction is done. The higher the number of features, the harder it gets to visualize the training

set and then work on it. Sometimes, many of these features are correlated or redundant. This is where dimensionality reduction algorithms come into play.

2. What is Dimensionality reduction

Dimensionality reduction is the process of reducing the number of random features under consideration, by obtaining a set of principal or important features.

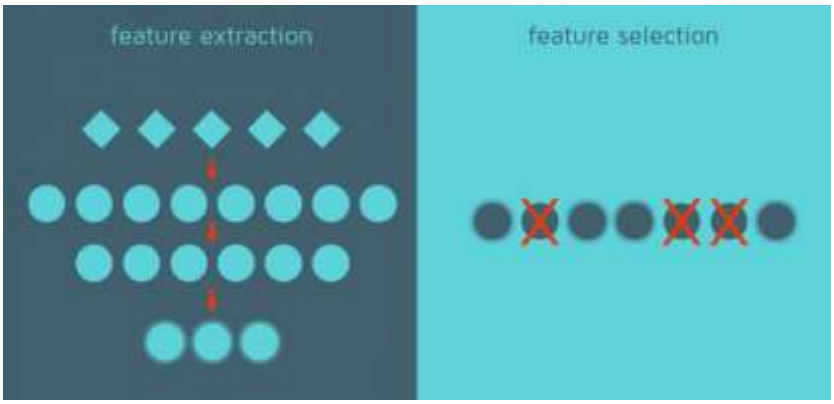
Dimensionality reduction can be done in 2 ways:

a. Feature Selection: By only keeping the most relevant variables from the original dataset

b. Feature Extraction: By finding a smaller set of new variables, each being a combination of the input variables, containing basically the same information as the input variables.

i. PCA(Principal Component Analysis)

ii. LDA(Linear Discriminant Analysis)



In this article, we will mainly focus on the Feature Extraction technique with its implementation in Python.

The feature Extraction technique gives us new features which are a linear combination of the existing features. The new set of features will have different values as compared to the original feature values. **The main aim is that fewer features will be required to capture the same information.** We might think that choosing fewer features might lead to underfitting but in the case of the Feature Extraction technique, the extra data is generally noise.

The feature extraction in machine learning technique provides us with new features, forming a linear combination of the existing ones. This results in a new set of features with values different from the original ones. The primary objective is to require fewer features to capture the same information. While one might assume that opting for fewer features could lead to underfitting, in the case of the feature extraction technique, the additional data is typically considered noise.

3. PCA(Principal Component Analysis)

In simple words, PCA is a method of obtaining important variables (in form of components) from a large set of variables available in a data set. It tends to find the direction of maximum variation (spread) in data. PCA is more useful when dealing with 3 or higher-dimensional data.

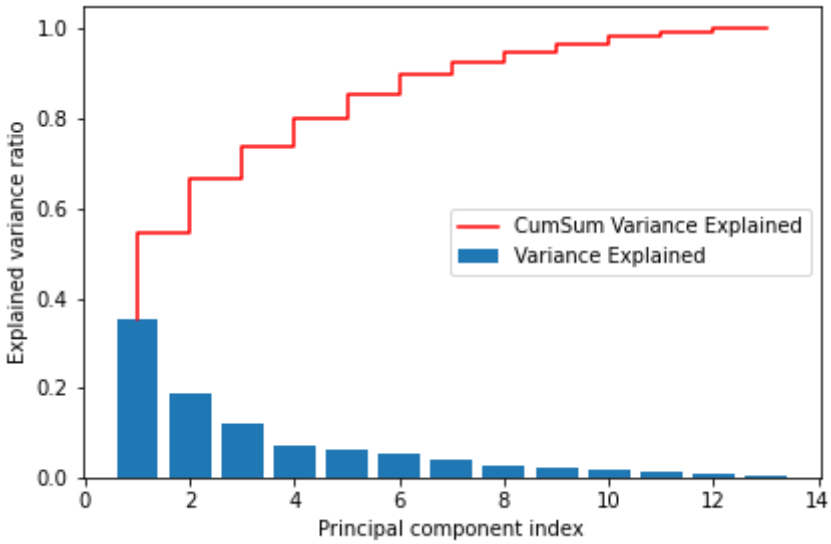
PCA can be used for anomaly detection and outlier detection because they will not be part of the data as it would be considered noise by PCA.Building PCA from scratch:

1. Standardize the data (X_std)
2. Calculate the Covariance-matrix
3. Calculate the Eigenvector & Eigenvalues for the Covariance-matrix.
4. Arrange all Eigenvalues in decreasing order.

5. Normalize the sorted Eigenvalues.

6. Horizontally stack the Normalized_ Eigenvalues =W_matrix

7. X_PCA=X_std .dot (W_matrix)



We can infer from the above figure that from the ***first 6 Principal Components*** we are able to capture 80% of the data. This shows us the Power of PCA that with only using 6 features we able to capture most of the data.

A principal component is a normalized linear combination of the original features in a data set.

The first principal component(PC1) will always be in the **direction of maximum variation** and then the other PC's follow.

We need to note that all the PC's will be **perpendicular** to each other. The main intention behind this is that no information present in PC1 will be present in PC2 when they are perpendicular to each other.

Semester : VI **Subject : Machine Learning** **Academic Year: 2023 - 2024**