DATA SCIENCE WITH PYTHON : DIMENSIONALITY REDUCTION #464

NAME: DEEPTHI M

SERIAL NUMBER: 172

BATCH NUMBER: 05

Dimension Reduction:

- Principal Component Analysis (PCA)
- Singular Value Decomposition (SVD)
- Linear Discriminant Analysis (LDA)

x1	x2	x3	x4	x5	x6	x7	x8

x1	x2	x3

Principal Component Analysis (PCA)

- **PCA** is a statistical method that is used to convert a set of correlated variables to a set of uncorrelated variables.
- It is a unsupervised machine learning model.
- Broadly, used as a EDA(Exploratory Data Analysis) method.

Used to convert large number of features into a fewer features with some loss of information:

- Identify the relationship between columns
- Visualize the multivariate data using 2 Principal Components
- It will help condense the features

Singular Value Decomposition (SVD)

The Singular Value Decomposition (SVD), a method from linear algebra that has been generally used as a dimensionality reduction technique in machine learning. SVD is a matrix factorisation technique, which reduces the number of features of a dataset

Linear Discriminant Analysis

Linear Discriminant Analysis or **Normal Discriminant Analysis** or **Discriminant Function Analysis** is a dimensionality reduction technique that is commonly used for supervised classification problems. It is used for modeling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.

When/Why to use :

•It is technique that is particularly useful in processing data where **multi-colinearity** exists between the **features/variables**.
•It can be used when **the dimensions of the input features are high** (e.g. a lot of variables).
•It can be also used for **denoising** and **data compression**.

Some of the Applications :

•Computation is fast
•Image compression

Dimensionality reduction

- Pros
  - reflects our intuitions about the data
  - allows estimating probabilities in high-dimensional data
    - no need to assume independence etc.
  - dramatic reduction in size of data
    - faster processing (as long as reduction is fast), smaller storage
- Cons
  - too expensive for many applications (Twitter, web)
  - disastrous for tasks with fine-grained classes
  - understand assumptions behind the methods (linearity etc.)
    - there may be better ways to deal with sparseness