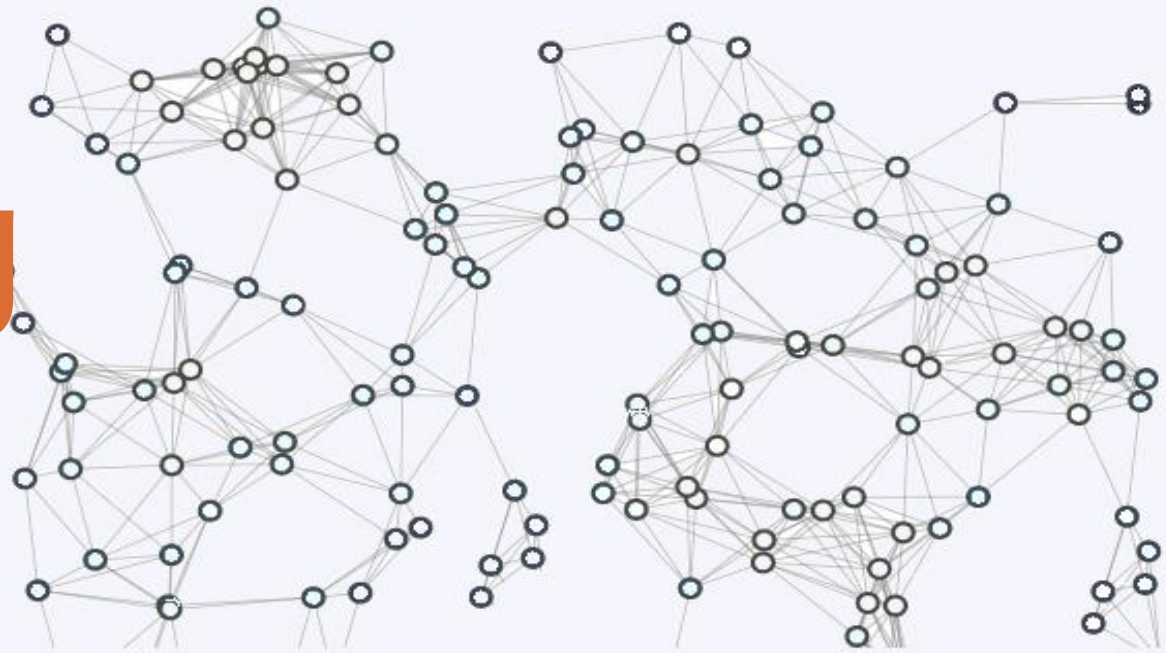
Data PreProcessing for Machine Learning



Rodolfo C. Raga Jr.

September 23-24, 2019

San Beda College

jundy.raga@gmail.com

Topic Outline

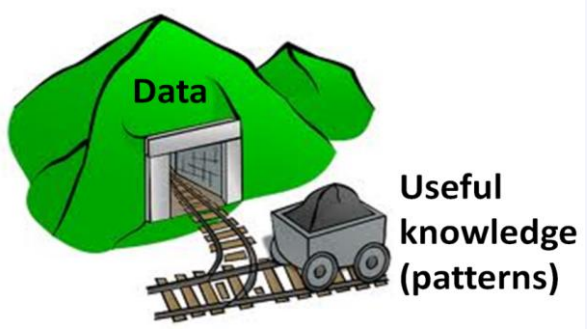
Day 1	Day 2
1) Intro to Machine Learning and Tensorflow **2) Data Preprocessing** 3) Learning Models (Regression) 3.1. NN Simple Linear 3.2. NN Multiple Linear 4) Model Training	1) Learning Models (Classification) 1.1. NN Logistic Regression 1.2. CNN Deep Learning 2) Model Testing 3) Model Evaluation / Validation Performance 4) Data Visualization

Introduction

- Machine Learning algorithms don’t work well with **raw data**. Before feeding such data to an ML algorithm, it must be **preprocessed**.

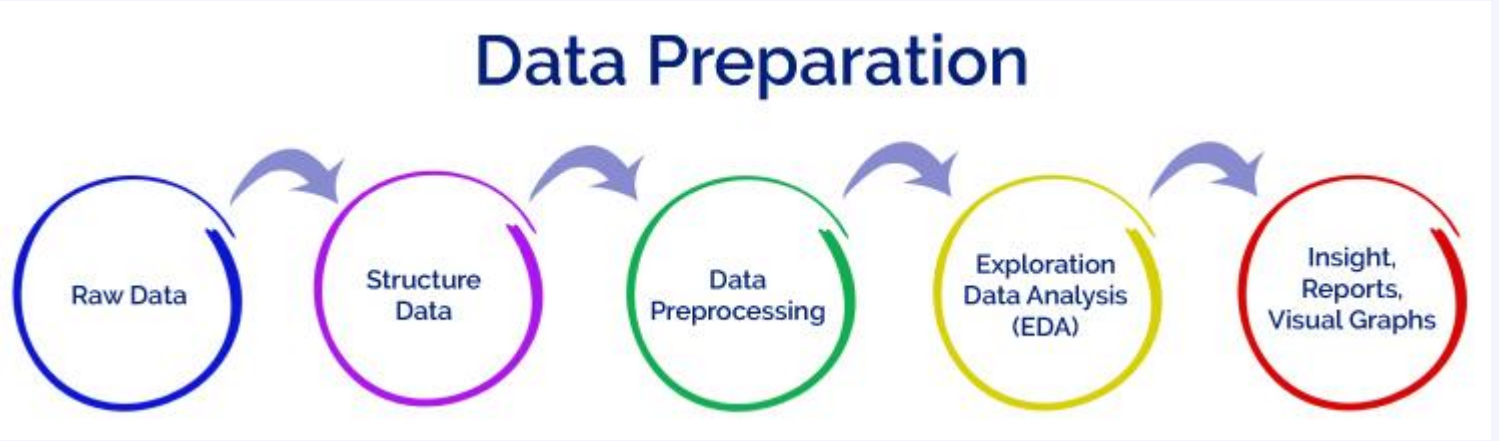
- **Pre-processing** refers to the transformations applied to raw data to transform it into clean data.

- **Data Preprocessing** requires application of techniques that converts the raw data into a clean

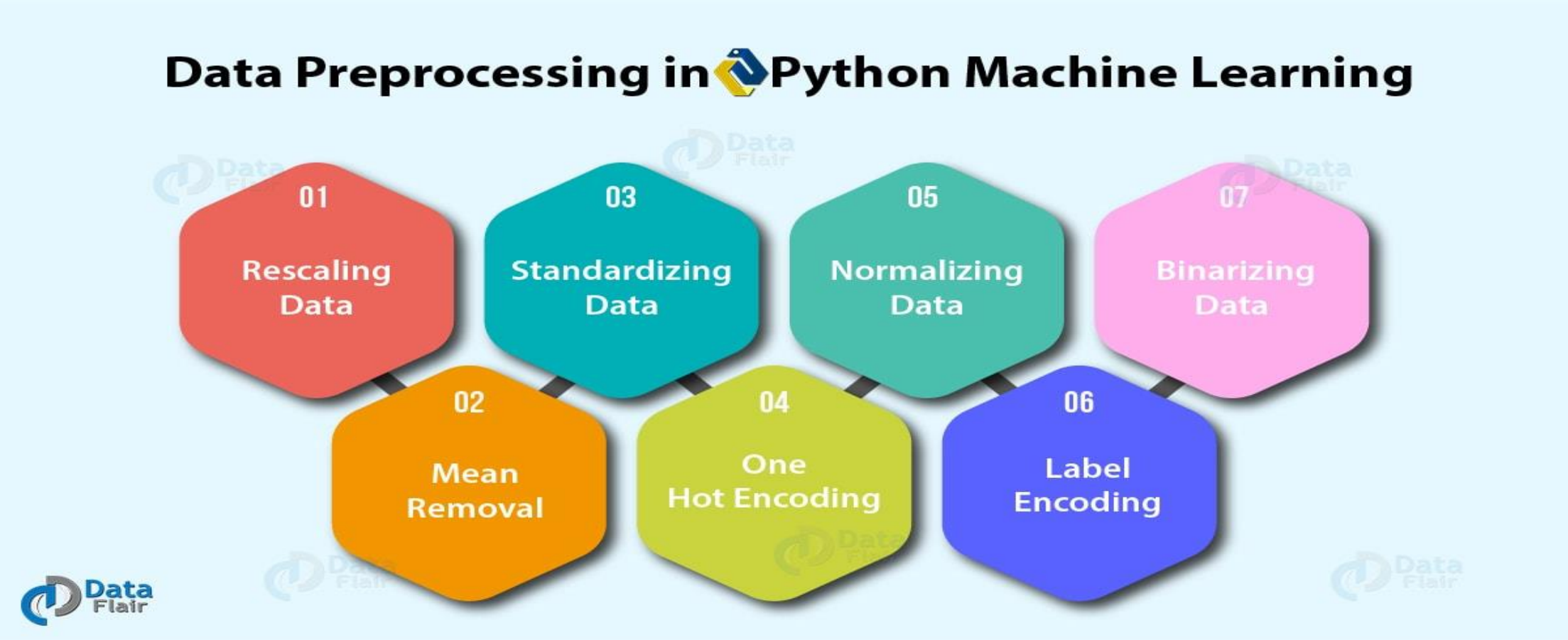


Introduction

- Need for Data Preprocessing
  - It can improve model performance
  - Many Machine Learning model require data input in a specific format.
  - Data set should be formatted in such a way that more than one Machine Learning and/or Deep Learning algorithms can work on them.



Major preprocessing techniques



Demo for effectiveness of Normalization

Data Pre-Processing techniques

- Rescale data
- Standardize data
- Binarize data
- Normalization

1. Rescaling Data

- Transforms a dataset so that its data features are rescaled to values between 0 and 1.
- Useful when data is comprised of attributes with varying scales
- Useful when using algorithms that weight inputs like regression and neural networks and with algorithms that use distance measures like K-Nearest Neighbors.
- Also improves the performance of optimization algorithms.

Rescaling Datasets using MinMaxScaler

- To rescale datasets, we first declare the MinMaxScaler library

  from sklearn.preprocessing import MinMaxScaler

- Syntax:

  MinMaxScaler(feature_range=(0, 1), copy=True)

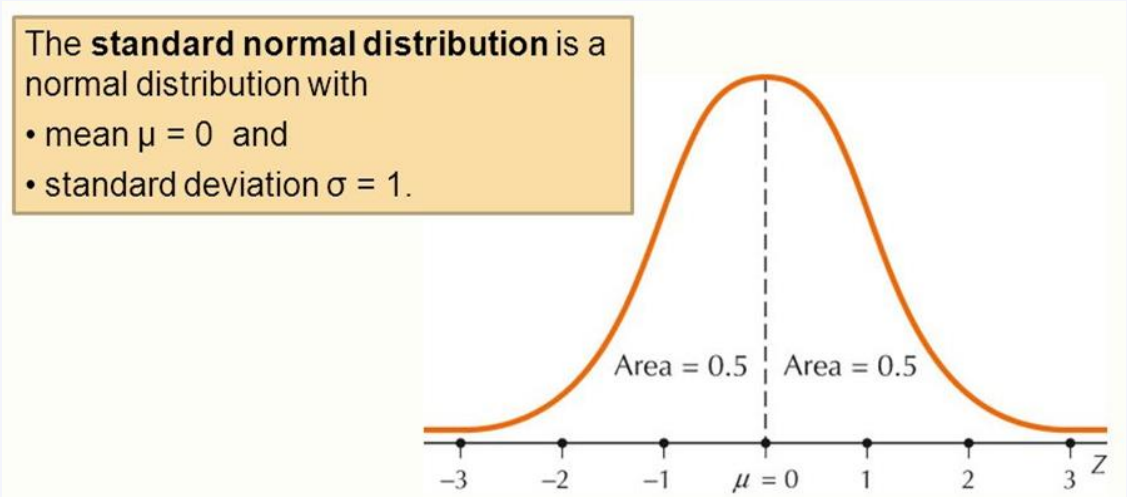
- Sample Use:

```
scaler = MinMaxScaler(feature_range=(0, 1))
rescaledX = scaler.fit_transform(X)
```

2. Standardizing Data

- It transforms the values in the dataset and shifts it so that the original mean value is placed at 0 and the standard deviation is 1.

- Gives data the property of a standard normal distribution (also known as Gaussian distribution).



Standardizing Datasets using scale

- To standardize datasets, we first declare the scale library

  from sklearn.preprocessing import scale

- Syntax:

  scale(data_to_scale)

- Sample Use:

  rescaledX2 = scale(X)

3. Binarize Data

- **Binarization** is the process of tresholding numerical features to get boolean values. Or in other words, assign a boolean value (True or False) to each sample based on a threshold.
- This is useful is useful as a feature engineering technique for creating new features that indicate something meaningful.
- The Binarizer class in sklearn implements binarization in a very intuitive way. The only parameters you need to specify are the threshold and copy. All values below or equal to the threshold are replaced by 0, above it by 1.

3. Binarizing Datasets using Binarizer

- The Binarizer class in sklearn implements binarization in a very intuitive way. The only parameters you need to specify are the threshold and copy. All values below or equal to the threshold are replaced by 0, above it by 1.
- Example:

```
from sklearn.preprocessing import Binarizer
binarizer = Binarizer(threshold=0, copy=True)
binarizer.fit_transform(X.f3.values.reshape(-1, 1))
```