**What is Data Preprocessing?**

It is a data mining technique that transforms raw data into an understandable format. Raw data(real world data) is always incomplete and that data cannot be sent through a model. That would cause certain errors. That is why we need to preprocess data before sending through a model.

**Why we use Data Preprocessing ?**

In Real world **data** are generally incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate **data**. Noisy: containing errors or outliers. Inconsistent: containing discrepancies in codes or names.

### Steps in Data Preprocessing

**Step 1 :** Import the libraries

**Step 2 :** Import the data-set

**Step 3 :** Check out the missing values

**Step 4 :** See the Categorical Values

**Step 5 :** Splitting the data-set into Training and Test Set

**Step 6 :** Feature Scaling

## HOW TO CODE STEP-1 & STEP 2

## Step 1: Import Libraries

First step is usually importing the libraries that will be needed in the program. A library is essentially a collection of modules that can be called and used. A lot of the things in the programming world do not need to be written explicitly ever time they are required. There are functions for them, which can simply be invoked.

A snippet of me importing the pandas library and assigning a shortcut “pd”.

**import pandas as pd**

## Step 2: Import the Dataset

A lot of datasets come in CSV formats. We will need to locate the directory of the CSV file at first (it’s more efficient to keep the dataset in the same directory as your program) and read it using a method called read\_csv ()which can be found in the library called pandas.

**# Read the data in the CSV file using pandas**

**df = pd.read\_csv(‘**creditcard.csv'**)**

**#print**

**print(df.head())**

**Year Datetime ... Home Team Initials Away Team Initials**

**0 1930.0 13 Jul 1930 - 15:00 ... FRA MEX**

**1 1930.0 13 Jul 1930 - 15:00 ... USA BEL**

**2 1930.0 14 Jul 1930 - 12:45 ... YUG BRA**

**3 1930.0 14 Jul 1930 - 14:50 ... ROU PER**

**4 1930.0 15 Jul 1930 - 16:00 ... ARG FRA**

**[5 rows x 20 columns]**

**print(df.shape)**

**(4572, 20)**

**print(df.index)**

**RangeIndex(start=0, stop=4572, step=1)**

**print(df.columns)**

**Index(['Year', 'Datetime', 'Stage', 'Stadium', 'City', 'Home Team Name',**

**'Home Team Goals', 'Away Team Goals', 'Away Team Name',**

**'Win conditions', 'Attendance', 'Half-time Home Goals',**

**'Half-time Away Goals', 'Referee', 'Assistant 1', 'Assistant 2',**

**'RoundID', 'MatchID', 'Home Team Initials', 'Away Team Initials'],**

**dtype='object')**

**Complete code-1(WorldCupMatches.csv)**

**#import library**

import pandas as pd

**# Read the data in the CSV file using pandas**

df = pd.read\_csv**("WorldCupMatches.csv")**

**#print first five**

print(df.head())

**#print shape(rows and columns)**

print(df.shape)

**#print index**

print(df.index)

**#print columns**

print(df.columns)

**Output of above code**

**Year Datetime ... Home Team Initials Away Team Initials**

**0 1930.0 13 Jul 1930 - 15:00 ... FRA MEX**

**1 1930.0 13 Jul 1930 - 15:00 ... USA BEL**

**2 1930.0 14 Jul 1930 - 12:45 ... YUG BRA**

**3 1930.0 14 Jul 1930 - 14:50 ... ROU PER**

**4 1930.0 15 Jul 1930 - 16:00 ... ARG FRA**

**[5 rows x 20 columns]**

**(4572, 20)**

**RangeIndex(start=0, stop=4572, step=1)**

**Index(['Year', 'Datetime', 'Stage', 'Stadium', 'City', 'Home Team Name',**

**'Home Team Goals', 'Away Team Goals', 'Away Team Name',**

**'Win conditions', 'Attendance', 'Half-time Home Goals',**

**'Half-time Away Goals', 'Referee', 'Assistant 1', 'Assistant 2',**

**'RoundID', 'MatchID', 'Home Team Initials', 'Away Team Initials'],**

**dtype='object')**

**Complete code-2(creditcard.csv)**

import pandas as pd

**# Read the data in the CSV file using pandas**

df = pd.read\_csv("creditcard.csv")

**#print**

print(df.head())

print(df.shape)

print(df.index)

print(df.columns)

**Output of above code**

**Time V1 V2 V3 ... V27 V28 Amount Class**

**0 0.0 -1.359807 -0.072781 2.536347 ... 0.133558 -0.021053 149.62 0**

**1 0.0 1.191857 0.266151 0.166480 ... -0.008983 0.014724 2.69 0**

**2 1.0 -1.358354 -1.340163 1.773209 ... -0.055353 -0.059752 378.66 0**

**3 1.0 -0.966272 -0.185226 1.792993 ... 0.062723 0.061458 123.50 0**

**4 2.0 -1.158233 0.877737 1.548718 ... 0.219422 0.215153 69.99 0**

**[5 rows x 31 columns]**

**(284807, 31)**

**RangeIndex(start=0, stop=284807, step=1)**

**Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',**

**'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',**

**'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',**

**'Class'],**

**dtype='object')**