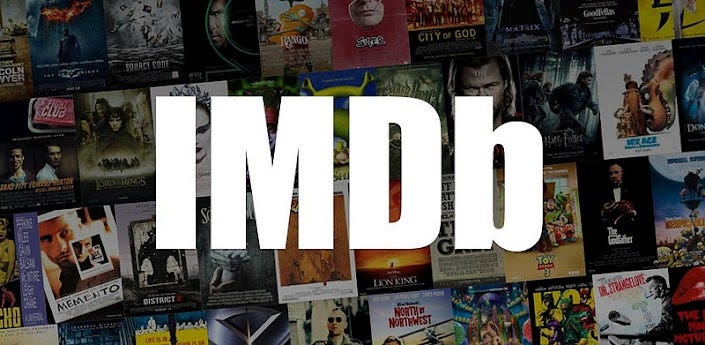
**Predicting IMDb Scores Using Machine Learning**

TEAM MEMBER

732521104054 : THIRISHA P

***Phase 3 Submission Document***

***Project : Predicting IMDb Scores***



**Introduction:**

* Predicting IMDb scores for movies or TV shows typically involves using machine learning models and features such as cast, crew, genre, user reviews, and more. You can use regression algorithms to build a predictive model.
* The quality of your predictions depends on the quality and quantity of data, as well as the choice of features and model.
* In this project , we will explore advanced regression techniques to enhance the accuracy and robustness of IMDb scores prediction models.
* Highlight the limitations of traditional linear regression models in capturing complex relationships.
* Emphasize the need for advanced regression techniques like Gradient Boosting and Neural Networks to enchance prediction accuracy.

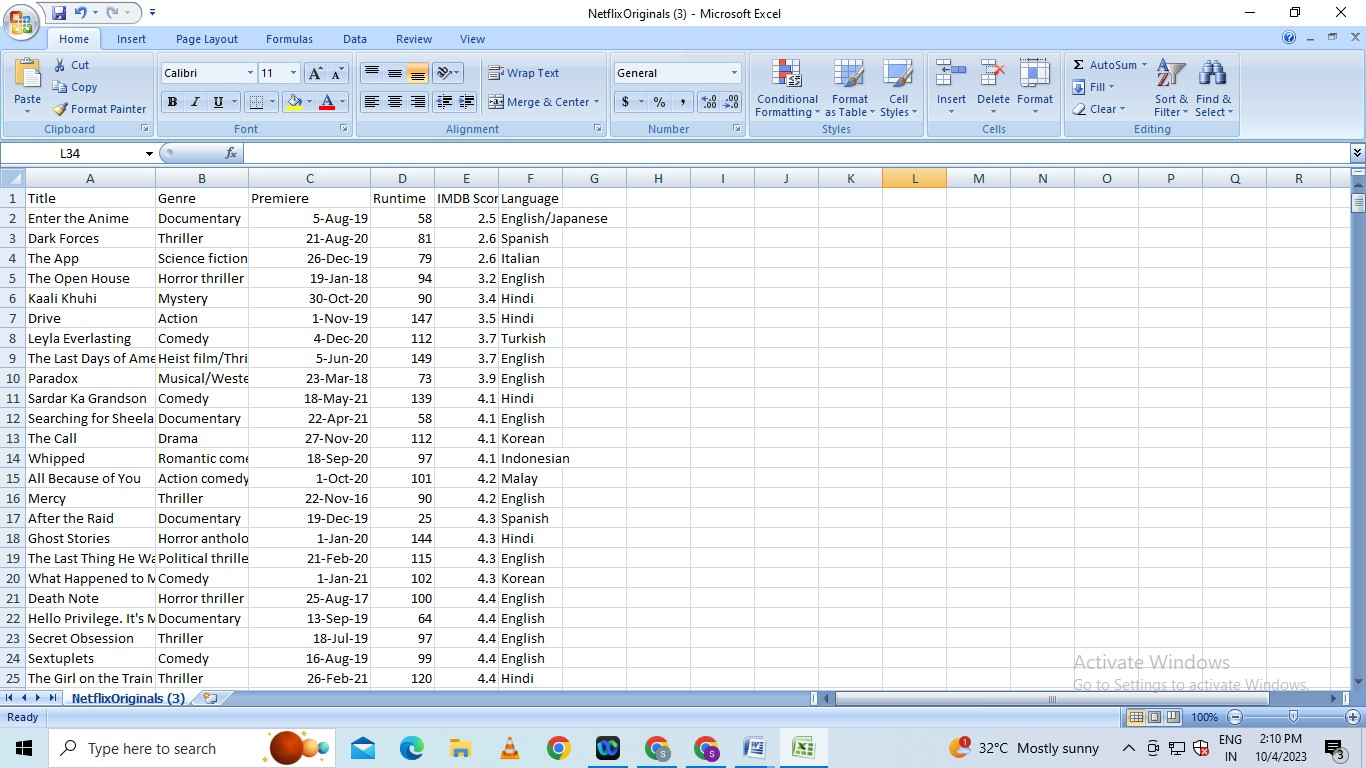
**Content For Project Phase 3 :**

Phase 3: Development Part 1

**Data Source :**

A Good Data for Predicting IMDb Scores using machine learning model should be Accurate , complete , accessible

**Dataset Link : (**[**https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores**](https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores)



**Dataloading**

# Pandas - DataFrame - Loading the dataset from various data sources

A dataset can be loaded from various data sources using relevant Pandas constructs (functions) as mentioned below:

* CSV file - read\_csv() function
* JSON file - read\_json() function
* Excel file - read\_excel() function
* Database table - read\_sql() function

All the above functions return a dataframe object and most of these functions have a parameter called 'chunksize'.

e.g. to load a JSON data file (myfile.json) you can use the below code

my\_df = pd.read\_json("myfile.json")

Here, my\_df is a pandas dataframe object.

chunksize - It is the number of rows(records) of the dataset (csv, excel, json, table, etc.) which you want to be returned in each chunk.

When you use this parameter - chunksize, these functions (read\_csv(), read\_sql(), etc.) return you an iterator which enable you to traverse through these chunks of data, where each chunk is of size as specified by chunksize parameter.

This 'chunksize' parameter is very useful when you are dealing with (loading) a large dataset and you have very limited memory (RAM) available on your machine. If 'chunksize' parameter is specified, only a chunk of data will be read into the dataframe at a time. Hence, if your specified chunksize is within your memory (RAM) limits, you can easily load large datasets using these constructs/functions of Pandas.

**INSTRUCTIONS**

Note: These instructions assume that you have completed the chapter on Numpy and have the necessary file housing\_short.csv in the correct directory. In case you don't have please go to the chapter of Numpy and complete the section on loading text file data first.

Please follow the below steps:

Please import pandas as pd

**Loading dataset from a CSV file**

(1) Please load the data from /cxldata/datasets/project/housing\_short.csv file by passing it to the read\_csv() function of Pandas library and store the returned dataframe in a variable called 'mydf'

**Data Preprocessing**

Data preprocessing in Machine Learning is a crucial step that helps enhance the quality of data to promote the extraction of meaningful insights from the data. Data preprocessing in Machine Learning refers to the technique of preparing (cleaning and organizing) the raw data to make it suitable for a building and training Machine Learning models. In simple words, data preprocessing in Machine Learning is a [data mining technique](https://www.upgrad.com/blog/most-common-examples-of-data-mining/) that transforms raw data into an understandable and readable format.

Data preprocessing steps are a part of the data analysis and mining process responsible for converting raw data into a format understandable by the ML algorithms.

Text, photos, video, and other types of unprocessed, real-world data are disorganized. It may not only be inaccurate and inconsistent, but it is frequently lacking and doesn’t have a regular, consistent design. Machines prefer to process neat and orderly information; they read data as binary – 1s and 0s.

So, it is simple to calculate structured data like whole numbers and percentages. But before analysis, unstructured data, such as text and photos, must be prepped and formatted with the help of data preprocessing in Machine Learning.

Now that you know what is data preprocessing in machine learning, explore the major tasks in data preprocessing.

## Data Preprocessing Steps In Machine Learning: Major Tasks Involved

Data cleaning, Data transformation, Data reduction, and Data integration are the major steps in data preprocessing.

### Data Cleaning

Data cleaning, one of the major preprocessing steps in machine learning, locates and fixes errors or discrepancies in the data. From duplicates and outliers to missing numbers, it fixes them all. Methods like transformation, removal, and imputation help ML professionals perform data cleaning seamlessly.

### Data Integration

Data integration is among the major responsibilities of data preprocessing in machine learning. This process integrates (merges) information extracted from multiple sources to outline and create a single dataset. The fact that you need to handle data in multiple forms, formats, and semantics makes data integration a challenging task for many ML developers.

### Data Transformation

ML programmers must pay close attention to data transformation when it comes to data preprocessing steps. This process entails putting the data in a format that will allow for analysis. Normalization, standardization, and discretisation are common data transformation procedures. While standardization transforms data to have a zero mean and unit variance, normalization scales data to a common range. Continuous data is discretized into discrete categories using this technique.

### Data Reduction

Data reduction is the process of lowering the dataset’s size while maintaining crucial information. Through the use of feature selection and feature extraction algorithms, data reduction can be accomplished. While feature extraction entails translating the data into a lower-dimensional space while keeping the crucial information, feature selection requires choosing a subset of pertinent characteristics from the dataset.

## Why Data Preprocessing in Machine Learning?

When it comes to creating a Machine Learning model, data preprocessing is the first step marking the initiation of the process. Typically, real-world data is incomplete, inconsistent, inaccurate (contains errors or outliers), and often lacks specific attribute values/trends. This is where data preprocessing enters the scenario – it helps to clean, format, and organize the raw data, thereby making it ready-to-go for Machine Learning models. Let’s explore various steps of data preprocessing in machine learning.

## Steps in Data Preprocessing in Machine Learning

 There are seven significant steps in data preprocessing in Machine Learning:

### ****1. Acquire the dataset****

Acquiring the dataset is the first step in data preprocessing in machine learning. To build and develop Machine Learning models, you must first acquire the relevant dataset. This dataset will be comprised of data gathered from multiple and disparate sources which are then combined in a proper format to form a dataset. Dataset formats differ according to use cases. For instance, a business dataset will be entirely different from a medical dataset. While a business dataset will contain relevant industry and business data, a medical dataset will include healthcare-related data.

### 2. Import all the crucial libraries

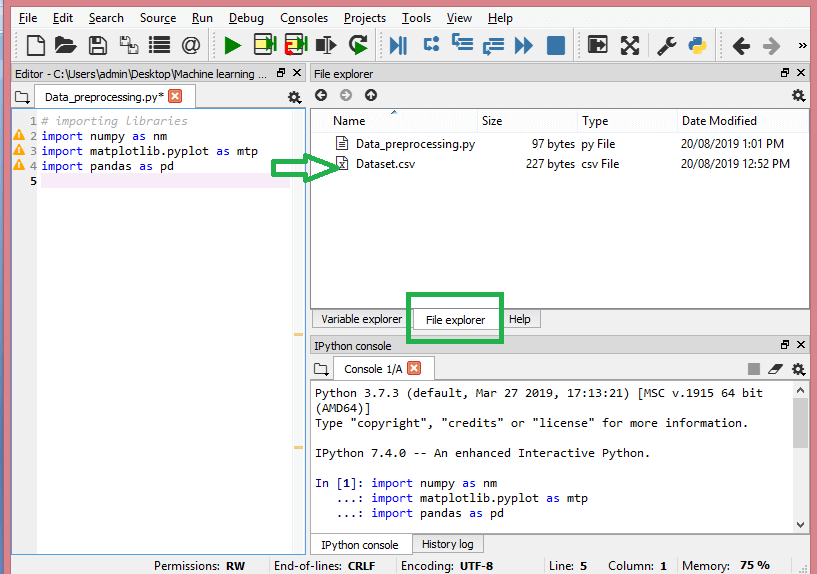
Since Python is the most extensively used and also the most preferred library by Data Scientists around the world, we’ll show you how to import Python libraries for data preprocessing in Machine Learning. Read more about [Python libraries for Data Science here.](https://www.upgrad.com/blog/python-libraries-for-data-science/) The predefined Python libraries can perform specific data preprocessing jobs. Importing all the crucial libraries is the second step in data preprocessing in machine learning. The three core Python libraries used for this data preprocessing in Machine Learning are:

* **NumPy** – NumPy is the fundamental package for scientific calculation in Python. Hence, it is used for inserting any type of mathematical operation in the code. Using NumPy, you can also add large multidimensional arrays and matrices in your code.
* **Pandas** – Pandas is an excellent open-source Python library for data manipulation and analysis. It is extensively used for importing and managing the datasets. It packs in high-performance, easy-to-use data structures and data analysis tools for Python.
* **Matplotlib** – Matplotlib is a Python 2D plotting library that is used to plot any type of charts in Python. It can deliver publication-quality figures in numerous hard copy formats and interactive environments across platforms (IPython shells, Jupyter notebook, web application servers, etc.).

### ****3. Import the dataset****

In this step, you need to import the dataset/s that you have gathered for the ML project at hand. Importing the dataset is one of the important steps in data preprocessing in machine learning. However, before you can import the dataset/s, you must set the current directory as the working directory. You can set the working directory in Spyder IDE in three simple steps:

1. Save your Python file in the directory containing the dataset.
2. Go to File Explorer option in Spyder IDE and choose the required directory.
3. Now, click on the F5 button or Run option to execute the file.



**How to extract the independent variables?**

To extract the independent variables, you can use “iloc[ ]” function of the Pandas library. This function can extract selected rows and columns from the dataset.

x= data\_set.iloc[:,:-1].values

In the line of code above, the first colon(:) considers all the rows and the second colon(:) considers all the columns. The code contains “:-1” since you have to leave out the last column containing the dependent variable. By executing this code, you will obtain the matrix of features, like this –

[[‘India’ 38.0 68000.0]

 [‘France’ 43.0 45000.0]

 [‘Germany’ 30.0 54000.0]

 [‘France’ 48.0 65000.0]

 [‘Germany’ 40.0 nan]

 [‘India’ 35.0 58000.0]

 [‘Germany’ nan 53000.0]

 [‘France’ 49.0 79000.0]

 [‘India’ 50.0 88000.0]

 [‘France’ 37.0 77000.0]]

**How to extract the dependent variable?**

You can use the “iloc[ ]” function to extract the dependent variable as well. Here’s how you write it:

y= data\_set.iloc[:,3].values

This line of code considers all the rows with the last column only. By executing the above code, you will get the array of dependent variables, like so –

array([‘No’, ‘Yes’, ‘No’, ‘No’, ‘Yes’, ‘Yes’, ‘No’, ‘Yes’, ‘No’, ‘Yes’],

      dtype=object)

## Conclusion

In conclusion, data preprocessing is an essential step in the data mining process and plays a crucial role in ensuring that the data is in a suitable format for analysis. This article provides a comprehensive guide to data preprocessing techniques, including data cleaning, integration, reduction, and transformation. Through practical examples and code snippets, the article helps readers understand the key concepts and techniques involved in data preprocessing and gives them the skills to apply these techniques to their own data mining projects. Whether you are a beginner or an experienced data miner, this article will provide valuable information and resources to help you achieve high-quality results from your data.