**Phase 3**

**Predicting IMDb Scores**

To load and preprocess a **IMDb**(Internet Movie Database) dataset for analysis, you can follow these general steps using Python and Pandas. Make sure you have a IMDb dataset in a suitable format available.

**1.Import Libraries :**

First, make sure you have the necessary libraries such as **numpy**, **pandas**, and **scikit-learn** installed, as they are commonly used for data manipulation and machine learning tasks.

**2. Load the IMDb Dataset :**

Load the **IMDb** dataset into a Pandas DataFrame. You can use **pd.read\_csv()** for CSV files, but the method may vary depending on the file format.

**handling the missing data :**

**Remove Missing Values (Not Recommended):**

* + The simplest approach is to remove rows with missing IMDb scores. However, this may result in a significant loss of data, especially if many entries have missing scores. Removing data should be a last resort because it can lead to biased result.
  + **Imputation:**
  + Imputation involves filling in missing values with estimates. There are several methods to consider:
  + **Mean or Median Imputation**: Replace missing scores with the mean or median score of the available data. This is a simple method but can introduce bias if the missing data is not missing at random.
  + **Regression Imputation**: Use regression models to predict IMDb scores based on other available data. For example, you might use features like the genre, director, or cast to predict scores.
    - **K-Nearest Neighbors (K-NN) Imputation**: Find k-nearest neighbors for a movie with missing data and use their IMDb scores to impute the missing value.

**encoding categorical data(one-bot):**

**Label Encoding:**

* Label encoding assigns a unique integer to each category. For example, you can assign a unique number to each genre or director. This method is suitable for ordinal categories where there is an inherent order, but it's not suitable for nominal categories (categories with no intrinsic order).

**One-Hot Encoding**:

* One-hot encoding is used for nominal categories where there is no inherent order. It creates binary columns for each category, and a "1" is placed in the column corresponding to the category to which the movie belongs.

**Binary Encoding**:

* Binary encoding is a compromise between label encoding and one-hot encoding. It converts each category into binary code and creates multiple binary columns.

**Frequency Encoding**:

* Frequency encoding involves replacing categories with the frequency of their occurrence in the dataset. This can capture the popularity or prevalence of each category.

**Splitting the data set into set and training :**

1. **Load Your Dataset**: Load your IMDb score dataset into a DataFrame. Ensure that your dataset contains a feature (independent variable) and a target variable (IMDb scores) that you want to predict or analyze.
2. **Split the Data**: Split your dataset into two parts: a training set and a testing set. The typical split ratio is 70-80% for training and 20-30% for testing. You can use scikit-learn's **train\_test\_split** function for this purpose:
3. **Feature Scaling and Preprocessing**: Before applying machine learning models, you might need to scale or preprocess your features. Common techniques include mean normalization and standardization. Be sure to apply the same transformations to both the training and testing data.
4. **Build and Train Your Model**: Using your training data (**X\_train** and **y\_train**), you can build and train your IMDb score prediction model. You can choose various machine learning algorithms, such as linear regression, random forests, or neural networks, depending on your task and dataset.



**Noconst :**

In the context of IMDb (Internet Movie Database), "nconst" is a unique identifier assigned to individuals involved in the film and television industry, such as actors, directors, writers, and other crew members. This identifier is used to reference and link various data and information about these individuals within the IMDb database.

**Primary Number :**

In IMDb (Internet Movie Database), there is no specific term called a "primary number" associated with an IMDb score. IMDb primarily uses a rating system called IMDb Ratings or IMDb Score, which is an average rating given to a movie or TV show by registered users of the platform.

**Birth year :**

In the context of IMDb (Internet Movie Database), the term "birth year" does not directly relate to the IMDb score. The IMDb score, also known as IMDb rating, is a numerical representation of a movie or TV show's popularity and quality based on user ratings and reviews. It is calculated by taking the average of ratings given by IMDb users.

**Death year :**

In the context of IMDb (Internet Movie Database), the "death year" refers to the year in which an individual, such as an actor, director, or any other person in the film and television industry, passed away or died. This information is often included in an individual's IMDb profile as part of their biographical details.

**Primary profession :**

In the context of IMDb (Internet Movie Database), an individual's "primary profession" is the primary area of the film and television industry in which they are primarily known for their work. IMDb allows individuals to list multiple professions or roles they have had in the industry, but it designates one as their "primary profession" to highlight their main area of expertise or recognition.

**know for title :**

In the context of IMDb (Internet Movie Database), the term "known for title" refers to a feature on the IMDb website that highlights the titles (movies or TV shows) for which a particular individual (actor, director, writer, etc.) is most recognized or known. This feature is displayed on the individual's IMDb profile and is based on the person's notable and well-regarded work in the industry.

**Libraries :**

In the context of IMDb score prediction, libraries typically refer to software libraries or frameworks used by data scientists, machine learning engineers, or researchers to develop and train predictive models for IMDb scores. These libraries provide tools, functions, and resources to work with data, build machine learning models, and evaluate their performance. Here are some key libraries and tools commonly used for IMDb score prediction:

1. **Python**: Python is a popular programming language for data science and machine learning. Many IMDb score prediction models are developed using Python due to its extensive ecosystem of libraries and tools.
2. **Scikit-Learn**: Scikit-Learn is a machine learning library in Python that provides a wide range of algorithms for tasks such as regression, classification, and model evaluation. It's often used to build IMDb score prediction models.
3. **Pandas**: Pandas is a data manipulation library in Python that helps with data preprocessing, cleaning, and transformation. It is useful for preparing IMDb data for analysis and model training.
4. **NumPy**: NumPy is another Python library that is crucial for numerical and array-based operations, making it essential for handling numerical data in IMDb score prediction.
5. **TensorFlow and PyTorch**: These deep learning frameworks are used for building more complex models, such as neural networks, which can be employed in IMDb score prediction tasks.
6. **XGBoost, LightGBM, and CatBoost**: These are gradient boosting libraries that excel in regression tasks. They can be applied to IMDb score prediction to create ensemble models that often perform well.
7. **Matplotlib and Seaborn**: These libraries are used for data visualization, which is important for understanding IMDb data, exploring relationships, and presenting results.
8. **Scrapy and Beautiful Soup**: These web scraping libraries can be used to gather IMDb data from the IMDb website for analysis and model training.
9. **Jupyter Notebooks**: Jupyter is an interactive development environment that allows data scientists to work with Python code, visualize data, and document their work. It's often used for exploratory data analysis and model development in IMDb score prediction projects.
10. **SQL Databases**: While not exactly libraries, SQL databases are sometimes used to store IMDb data and conduct SQL queries to retrieve and manipulate data for analysis.

**Feature Scaling :**

**1.Standardization (Z-score normalization)**: Standardization scales your features to have a mean of 0 and a standard deviation of 1. This transformation is suitable for most algorithms, and it doesn't assume that your data follows a particular distribution.

**2.Min-Max Scaling (Normalization)**: Min-Max scaling scales your features to a specific range, often between 0 and 1. It's useful when you want to preserve the original data's distribution and make it more interpretable.

**3.Robust Scaling**: Robust scaling is a variation of Min-Max scaling that is less sensitive to outliers. It uses the interquartile range (IQR) to scale data.

**4.Log Transformation (for positive-skewed data)**: If your IMDb score data is positively skewed (right-skewed), you can apply a logarithmic transformation to make it more normally distributed. This can help stabilize the variance and make the data more amenable to standardization.