**EXPERIMENT 3**

**AIM**: To preprocess dataset using different preprocessing techniques.

**TO DO**:-

1. Data Cleaning - removing missing values(demonstrate removing and replacing Null values)

2. Data Cleaning - removing noisy values(Binning technique), removing outliers- Interquartile Range Method,Boxplot

3. Data Transformation - converting numerical attributes to categorical and vice versa/ one hot encoding

4. Data Transformation - data normalization(Z- score transformation)

5. Data Reduction - reducing the number of rows by attribute-oriented induction or numerosity reduction

**THEORY:**

Data preprocessing, a component of data preparation, describes any type of processing performed on raw data to prepare it for another data processing procedure. It has traditionally been an important preliminary step for the data mining process. More recently, data preprocessing techniques have been adapted for training machine learning models and AI models and for running inferences against them.

Data preprocessing transforms the data into a format that is more easily and effectively processed in data mining, machine learning and other data science tasks. The techniques are generally used at the earliest stages of the machine learning and AI development pipeline to ensure accurate results.

There are several different tools and methods used for preprocessing data, including the following:

1. sampling, which selects a representative subset from a large population of data;
2. transformation, which manipulates raw data to produce a single input;
3. denoising, which removes noise from data;
4. imputation, which synthesizes statistically relevant data for missing values;
5. normalization, which organizes data for more efficient access; and
6. feature extraction, which pulls out a relevant feature subset that is significant in a particular context.

**Numpy:**

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software.

This is the foundation on which almost all the power of Python’s data science toolkit is built, and learning NumPy is the first step on any Python data scientist’s journey.

Features:

* Provides fast, precompiled functions for numerical routines
* Array-oriented computing for better efficiency
* Supports an object-oriented approach
* Compact and faster computations with vectorization

**Pandas:**

Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

Features:

* It has a fast and efficient DataFrame object with the default and customized indexing.
* Used for reshaping and pivoting of the data sets.
* Group by data for aggregations and transformations.
* It is used for data alignment and integration of the missing data.

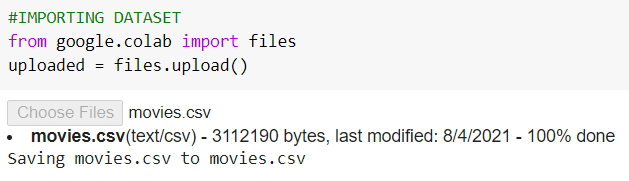
**IMPLEMENTATION:**

1. **Load data in Pandas.**

from google.colab import files

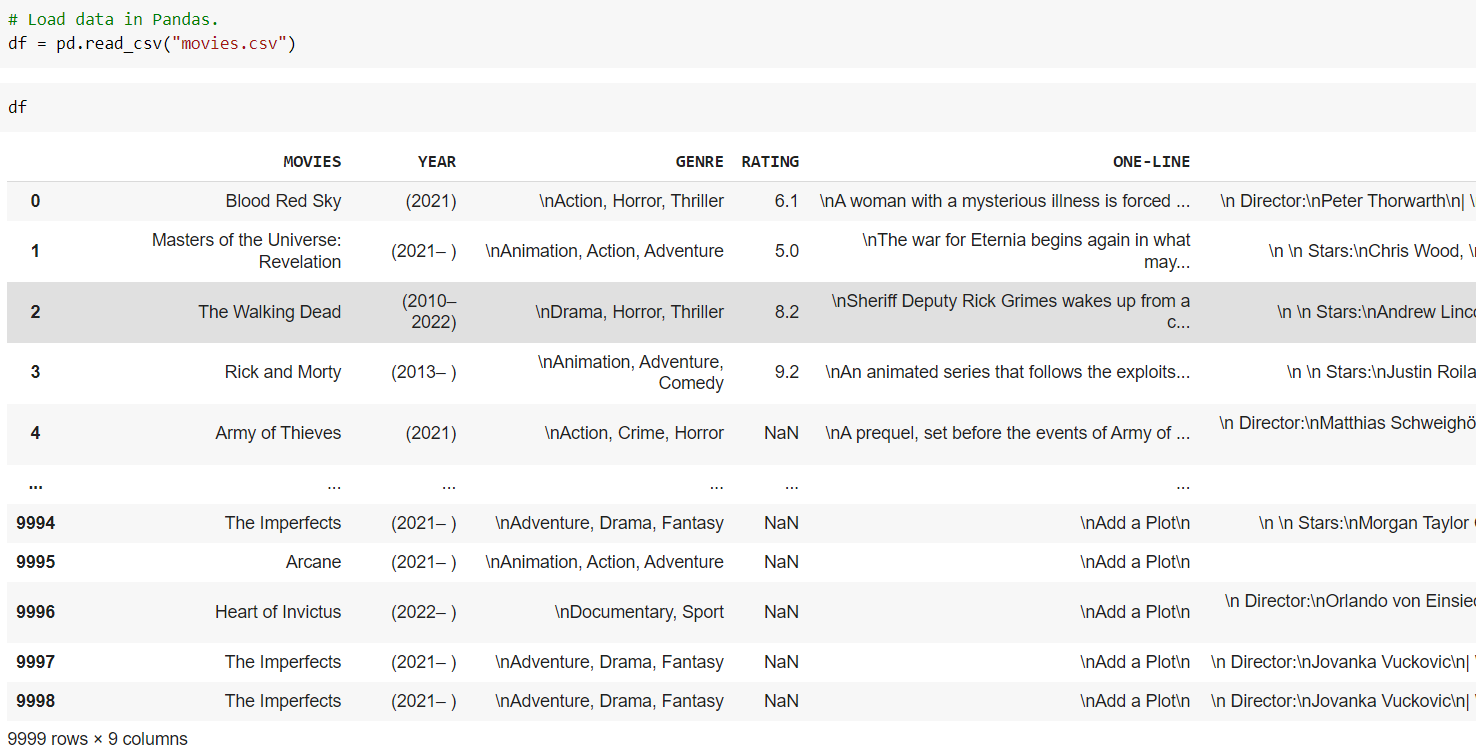
uploaded = files.upload()

>>The above command will enable us to upload the dataset file in the google collab notebook.



pd.read\_csv()

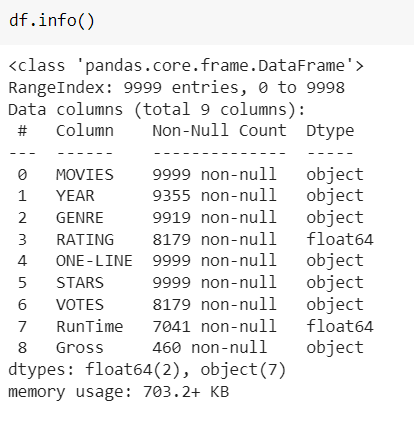
>>The following command will load the data in pandas and will show us some rows and columns from our dataset.



1. **Description of the dataset.**

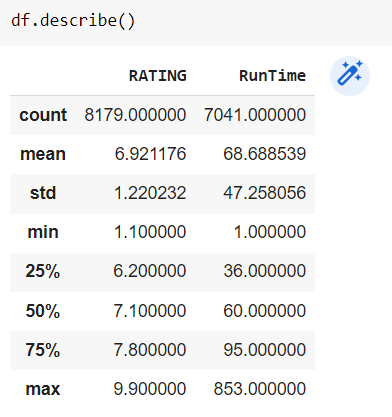
df.info()

>>This method prints information about a DataFrame including the index data type and columns, non-null values and memory usage.



df.describe()

>>The describe() method returns a description of the data in the DataFrame. If the DataFrame contains numerical data, the description contains this information for each column: count - The number of not-empty values. mean - The average (mean) value.



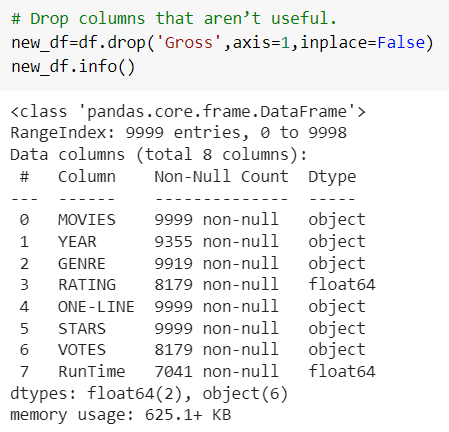
1. **Drop columns that aren’t useful.**

df.drop()

>>The drop() function is used to drop specified labels from rows or columns. Remove rows or columns by specifying label names and corresponding axis, or by specifying directly index or column names.

labels >> Index or column labels to drop.

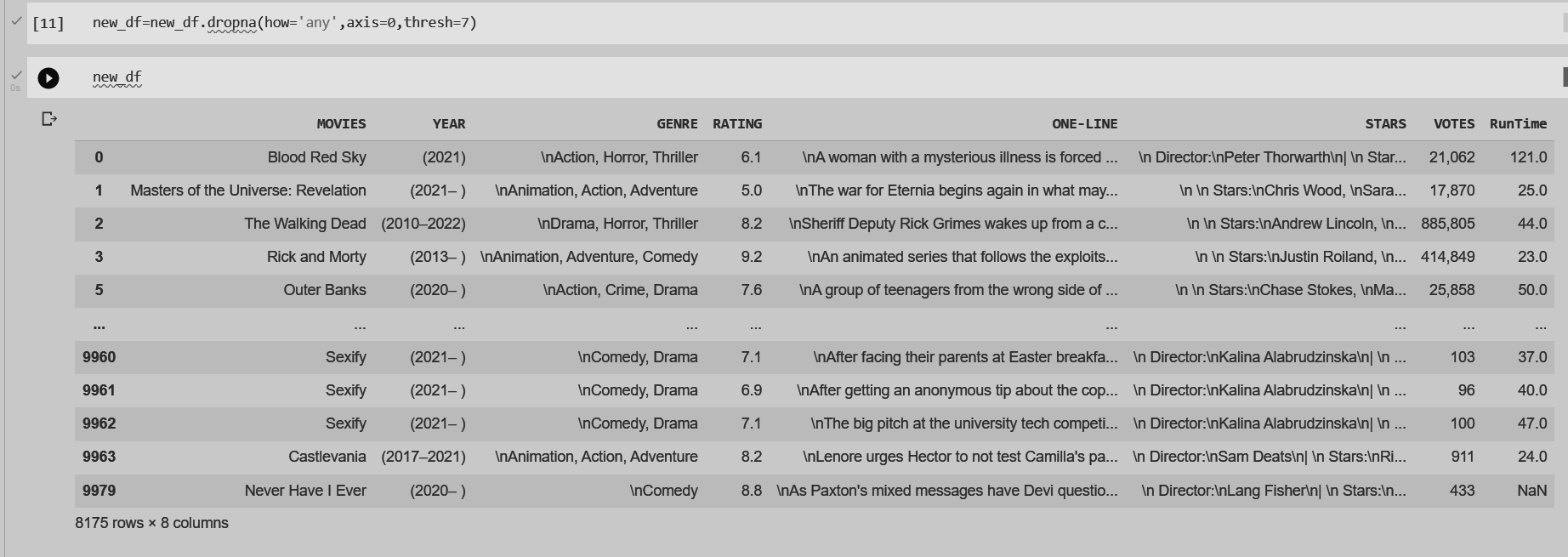
axis >> Whether to drop labels from the index (0 or ‘index’) or columns (1 or ‘columns’).



1. **Drop rows with maximum missing values.**

newdf=df.dropna(how='any')

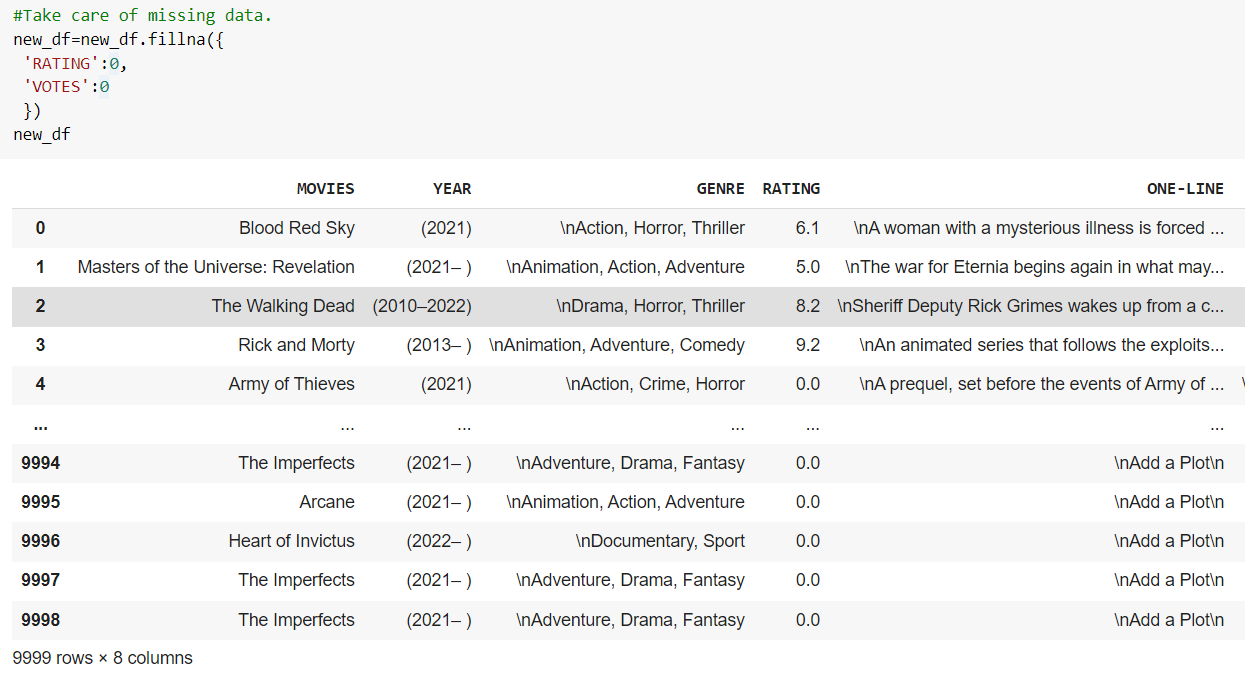
>> The dropna() method removes the rows that contain NULL values. The dropna() method returns a new DataFrame object unless the inplace parameter is set to True, in that case the dropna() method does the removing in the original DataFrame instead.



1. **Take care of missing data.**

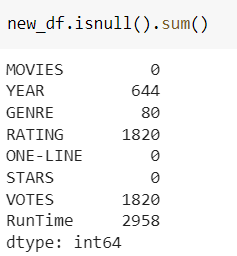
new\_df=df.fillna(0)

>> In pandas, the Data Frame provides a method fillna()to fill the missing values or NaN values in DataFrame. Whatever value you put in the () will be filled in the missing values



new\_df.isnull().sum()

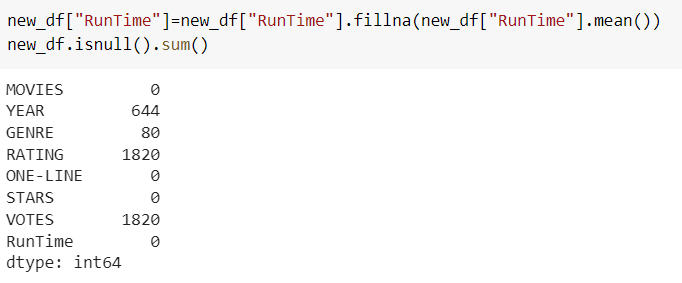
>> This will find the total number of null items in each column.



new\_df["RunTime"]=new\_df["RunTime"].fillna(new\_df["RunTime"].mean())

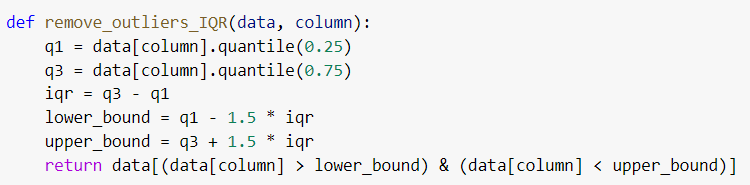
new\_df.isnull().sum()

>> This will replace all the null values in RunTime with the mean of all the values in the column RunTime.

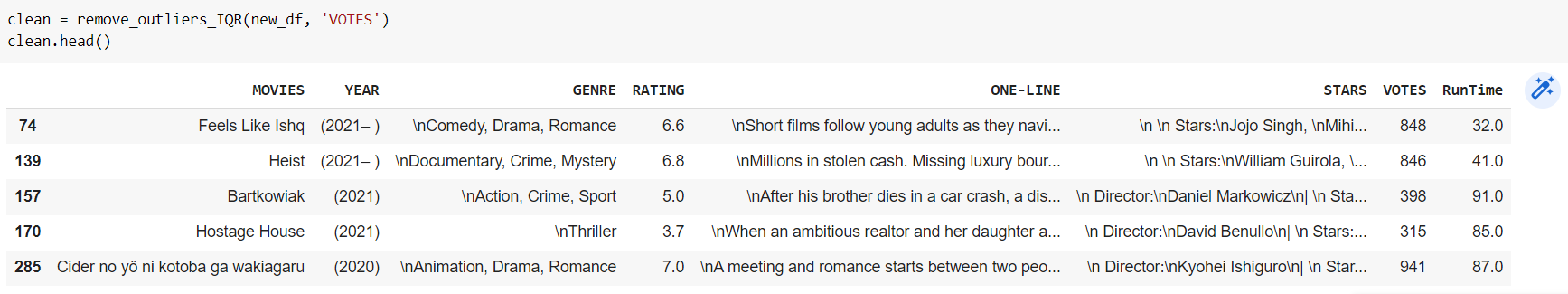


1. **Removing outliers using Interquartile Range Method**

>>function to return the outliers

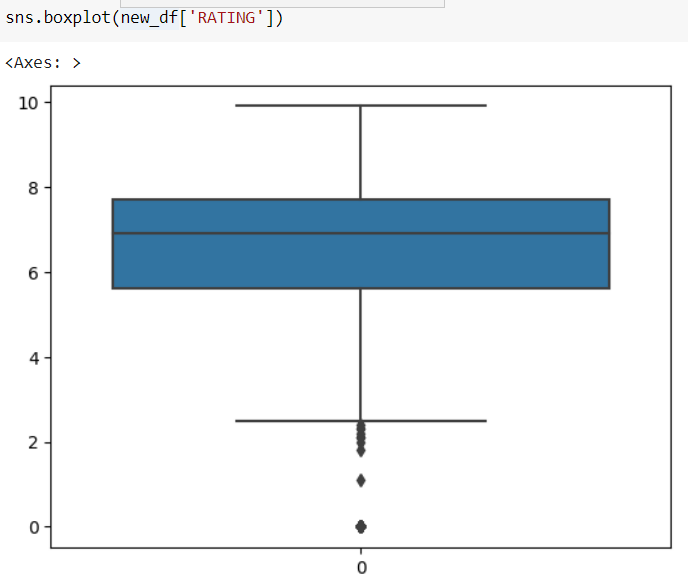
****

>> These returned values will have to be dropped from the dataset

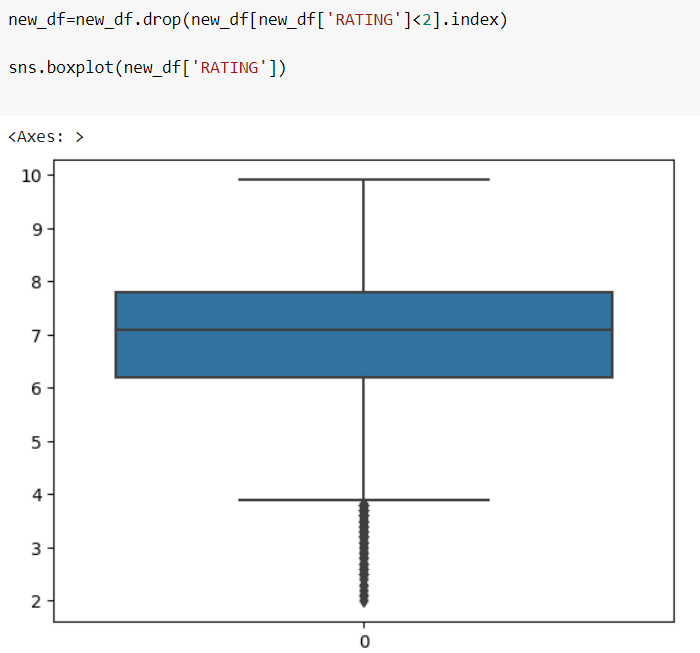


1. **Removing outliers using Boxplot method**

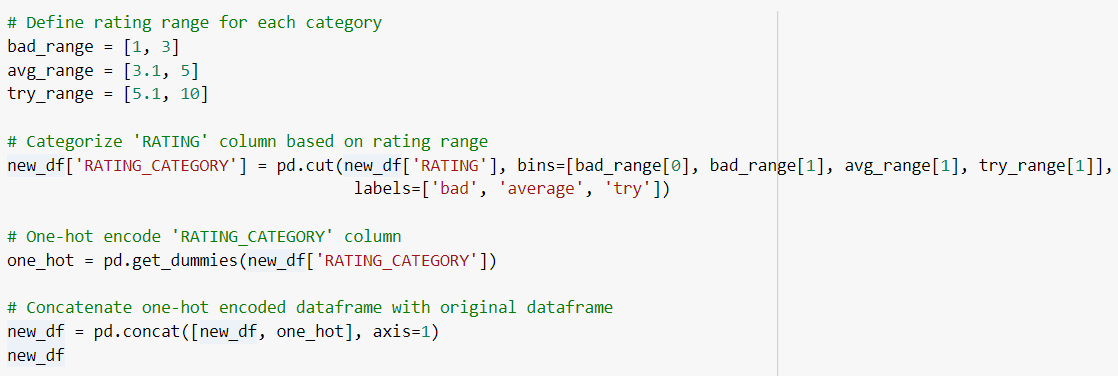
>> Mapping the box plot

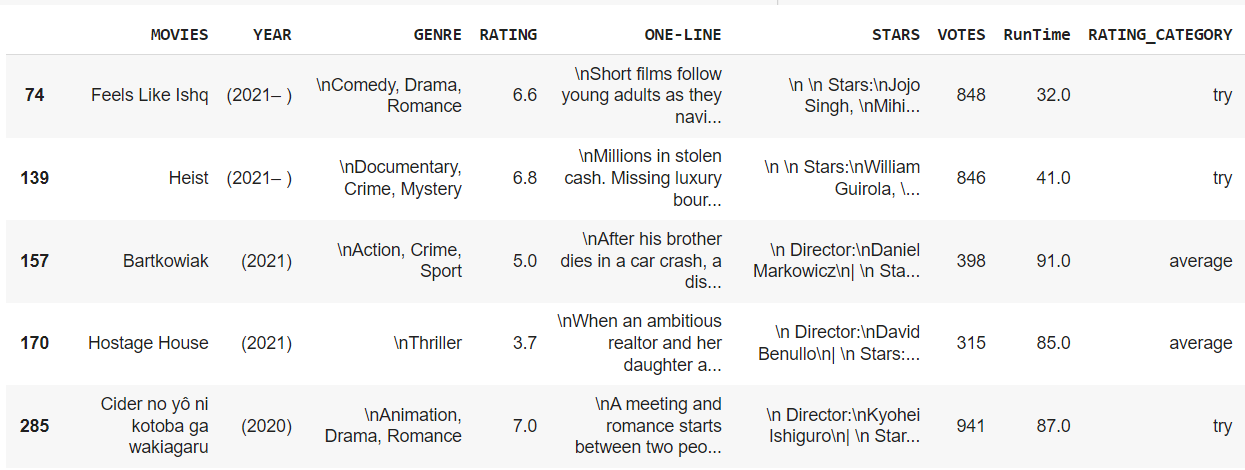
****

>> Removing the outliers that are less than 2



1. **Categorizing numerical values into groups using Hot Encoding**

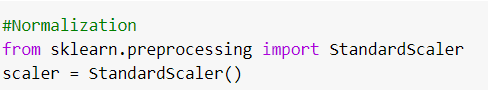
****

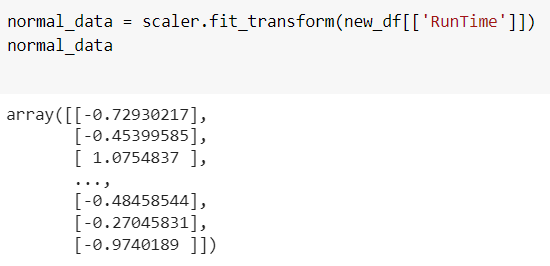
****

1. **Normalization (Z- score transformation)**

Z-score normalization (also known as standardization) is a method of normalizing numerical data. It involves transforming the data such that it has a mean of zero and a standard deviation of one.

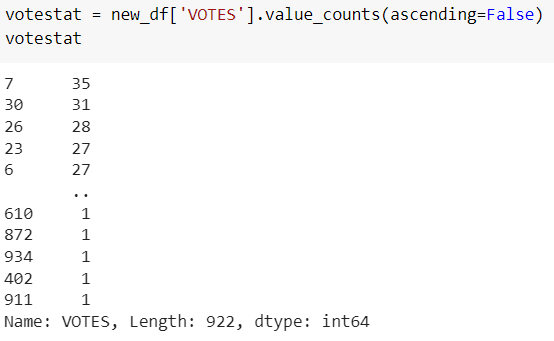
>> create a StandardScaler object 'scaler' from the sklearn.preprocessing module.

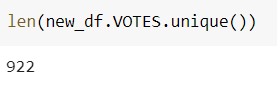


>>fit and transform the 'RunTime' column of the 'new\_df' using the scaler by calling the fit\_transform() method on the column.

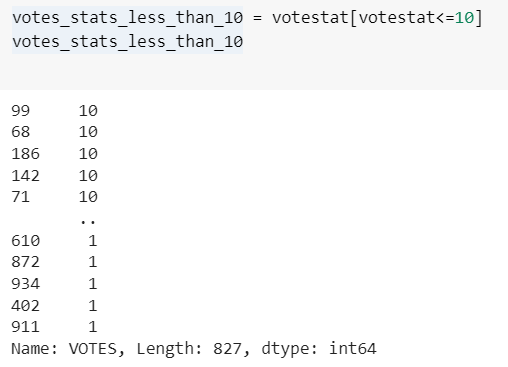
1. **Data Reduction**

>> Calculates the frequency count of unique values in the 'VOTES' column

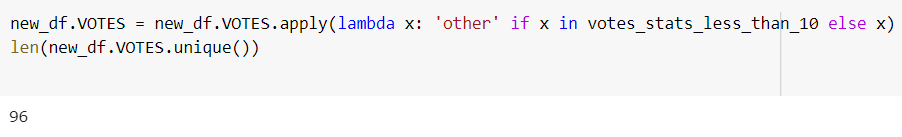
****

****

>>create a new pandas series 'votes\_stats\_less\_than\_10' that contains only the frequency counts of unique values in the 'VOTES' column of the pandas dataframe 'new\_df' that are less than or equal to 10.

****

>> Replace all the unique values in the 'VOTES' column that have a frequency count less than or equal to 10 with the string 'other'.

****

**CONCLUSION:**

Hence, we’ve successfully performed various techniques of preprocessing on our dataset. These techniques included handling missing values, outliers, one hot encoding, z-score transformation and data reduction.