**Exploratory Data Analysis using Python**

**Table of content**

1. Introduction
2. What is Exploratory Data Analysis?
3. Need of Exploratory Data Analysis
4. Objectives of Exploratory Data Analysis
5. Steps involved in Exploratory Data Analysis
6. Tools for Exploratory Data Analysis
7. Books Exploratory Data Analysis
   1. Purpose
   2. Data Collection
      1. Books Data
      2. Ratings Data
      3. Users Data
         1. Data Synthesis
         2. Explanation of data synthesis
   3. Data Integration
   4. Data description / Data understanding
   5. Check and Fill null values
      1. Find null values
      2. Fill null values
   6. Handle Outliers
      1. What are outliers?
      2. Why outliers are bad?
      3. Find outliers
      4. Remove outliers
   7. Data binning / data bucketing
      1. Importance of data binning
   8. Understanding relationships and new insights using plots
      1. Seaborn
      2. Matplotlib
      3. Correlation matrix
   9. Data visualization
      1. Distribution plot graph
      2. Boxplot visualization
      3. Correlation matrix
      4. Bar plot
      5. Rating Count Distribution plot
   10. Create random values column
       1. create random values of book category with specific ratios.
       2. Create random values of gender column with specific ratios.
       3. Concatenate gender and category data with dataset.
       4. Count plot
       5. Creating a count plot of Gender column.
       6. Creating a count plot of DBookCategory.

**1.Introduction**

John Tukey introduced [Exploratory Data Analysis (EDA](http://www.edureka.co/blog/exploratory-data-analysis-in-python/) with Python in the 1970s. EDA is the first step of data analysis, in which statistics and probability are used to identify trends and get a better understanding of a data set.

Exploratory data analysis is a strategy for understanding many elements of data, such as identifying null values and filling them in, locating redundant values and cleaning them up, locating significant variables, and removing all superfluous noise in data that could compromise data accuracy.

**2.What is Exploratory Data Analysis (EDA)?**

Exploratory Data Analysis is a process of evaluating data in order to summarize its primary properties, which is usually done using visual approaches.

To move on to more complicated processes in the data processing life cycle, drive conclusions on data insights for conclusive interpretations.

***Example:***

You are going to make analysis of sales data. You will make decisions on statistics. You will attempt to investigate the summary of data, such as the total number of records, unrelated records, inconsistencies, and outliers, as well as strategies for cleaning data and preparing to train a Machine Learning Model.

**3.Need of Exploratory data analysis:**

An important step before building a machine learning model on data is exploratory data analysis. After EDA, you'll know whether the selected features are adequate enough to develop a model or whether there's any data correlation, and you may either return to data preprocessing or move on to data modelling.

Build a supervised, unsupervised, or semi-supervised machine learning model for predictions once the EDA process is completed and its features are drawn.

After finishing the EDA process, you will have a better grasp of data, as well as a variety of graphs, heat maps, and data correlations that are simple to comprehend.

**4.Objectives of Exploratory Data Analysis**

* Make certain the data is free of errors.

(All data, including null values and other anomalies, is devoid of all tendency.)

* Identify and simply eliminate defective spots.
* Recognize the connections between variables.

**5.Steps involved in EDA**

* Description / understanding of data.

Helps to understand what kind of data, how many rows, columns and check the statistics of data.

* Handling missing values.

Find the null values in data sets and fill them, also clean redundancy that are not necessary for making conclusions or interpretations.

* Handling outliers

Remove outliers that cause noise in data. It may cause over fit or under fit the model when working on model building.

* Understanding relationships and new insights through plots.

Define the relationships between variables of data and visualize the data using different graphs.

**6.Tools for Exploratory Data Analysis**

There are many tools for Exploratory Data Analysis for data visualization and data cleaning. Some of them are Excel, Tableau, Weka etc.

But EDA using python most useful packages are Pandas, Numpy, Matplotlib and Seaborn.

**7.Books Exploratory Data Analysis**

***7.1 Purpose:***

The goal of EDA on books data is to better understand the data, find data statistics, and verify the total number of records and user ratings. For a better understanding, find the data correlation and plot various graphs. Find and eliminate null values, inconsistencies, and outliers.

***7.2 Data Collection:***

The practice of acquiring and measuring information on variables of interest is known as data collection.

Maintaining the integrity of research, making educated business decisions, and guaranteeing quality assurance all need accurate data collecting.

In this project of Books exploratory data analysis, collect dataset from IIF (INSTITUT FOR INFORMATIK FREIBURG) link ( <http://www2.informatik.uni-freiburg.de/~cziegler/BX/>).

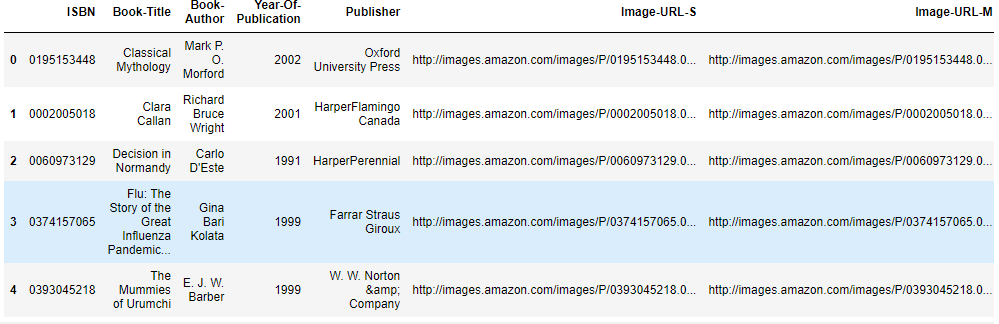
The data consists of three different parts: books data, user’s data and ratings data.

***7.2.1 Books data***

In books dataset, all data related to books like ISBN, title, author and year of publication.

Total records in books data set 271360 and total fields 8.

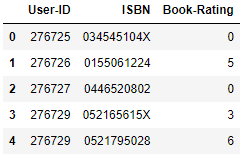
books=pd.read\_csv(r"C:\Users\Saddique\Downloads\bx\BX-Books.csv", sep=";", encoding='latin\_1', error\_bad\_lines=**False**)



**7.2.2** ***Ratings Data***

In ratings data Contains Rating, ISBN and User ID.

book\_rating=pd.read\_csv(r"C:\Users\Saddique\Downloads\bx\BX-Book-Ratings.csv", sep=";")



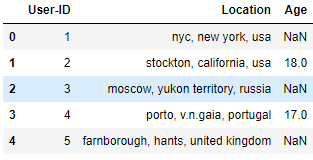
***7.2.3 Users data***

In rating data, following are columns User ID, Location, age.

Read user data

users=pd\_read\_csv(r"C:\Users\Saddique\Downloads\bx\BX-Users.csv", sep=";")

Display



***7.3 Data Integration***

Merge books and user data on ISBN.

rating\_with\_books=book\_rating.merge(books, on="ISBN")

Then, merge rating with books data with user’s data on User ID.

user\_rating\_books=users.merge(rating\_with\_books, on="User-ID")

***7.4 Data Description / Data Understanding***

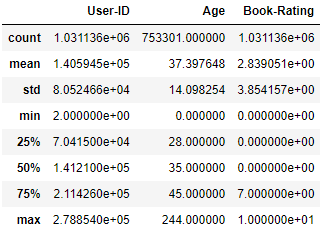
Statistical approaches such as mean, median, and mode percentile can be used to examine and comprehend data. Find the total number of records and check the data fields. In data, there are fields of numeric and string data types.

Check the total record of data set. Find total number of rows and columns. Total 12 columns and 1031136 rows.

user\_rating\_books.shape

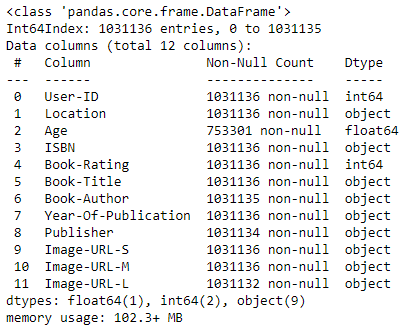
(1031136,10)

Check the statistics of data (Only numerical columns) shows standard deviation, mean value minimum and maximum values of each numeric column.



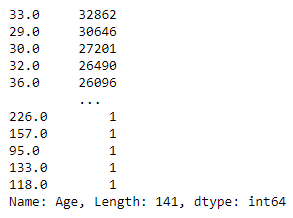
Check the total information of records as column names, data types and total records in each field.

user\_rating\_books.info()



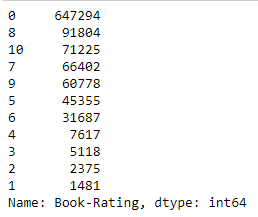
Count the unique values of “Age” columns. Use value counts() function to count total values of same age users.

user\_rating\_books['Age'].value\_counts()



Count the unique values of Book\_Rating column. Use value counts() function to count total values of same age users.

user\_rating\_books['Book-Rating'].value\_counts()



***7.5 Check and fill null values***

When no information is provided for one or more components, or for a whole unit, it is known as [missing data](https://www.analyticsvidhya.com/blog/2021/05/dealing-with-missing-values-in-python-a-complete-guide/). In real-life circumstances, missing data is a major issue. In pandas, missing data is also referred to as NA (Not Available) values. Many datasets come in DataFrame with missing data, either because it exists but was not collected or because it never existed. Assume that different people being questioned choose not to reveal their income, and that other users choose not to give their address, and that as a result, several datasets are missing. Null valus means nothing or unknown value in given field.

**It has some disadvantages.**

You can't accurately anticipate data with missing values since your results will be altered, and you won't be able to predict the values reliably.

Misleading data can make data analysis time-consuming and error-prone. The outcomes could be deceiving.

**7.5.1 Check the null values from data set.**

Check the null values of “Book Rating” column. But there was not any single null value.

user\_rating\_books['Book-Rating'].isnull().sum()

0

Check the null values of “Age” column. Total 277835 rows are empty/ Null values.

user\_rating\_books['Age'].isnull().sum()

277835

**7.5.2** ***Fill null values***

Fill up the blanks with integer or float data types; this data type has two isbn13 and original publication year fields.

Fill the missing values of “isbn13” field.

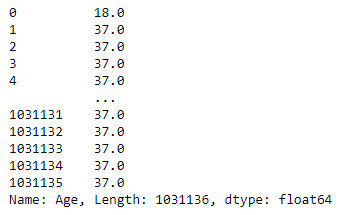
The fillna () method in Python is commonly used to fill null values. It returns a float data type, thus use the int () method to convert it to an integer. Then, in the original field "Age," map the result.

user\_rating\_books['Age'].mean()

37.39764848314286

user\_rating\_books['Age']=user\_rating\_books['Age'].fillna(int(user\_rating\_books['Age'].mean()))

user\_rating\_books['Age']



**7.6 Handle Outliers:**

***7.6.1 What are outliers?***

[Outliers](https://www.geeksforgeeks.org/detect-and-remove-the-outliers-using-python/) are records that stand out from the others in a field. Anomalies in the results of algorithms and analytical systems are caused by outliers.

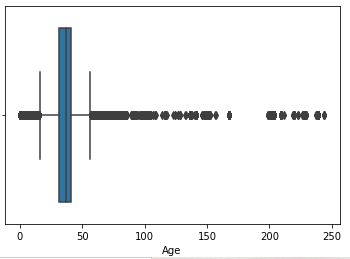
***7.6.1 Why outliers are bad?***

Outliers are exceptional numbers in a dataset that can cause statistical analysis to be skewed and assumptions to be broken. Outliers increase data variability, which reduces statistical power.

***7.6.3 Find outliers***

Use boxplot to find outliers from fields. Box plot indicates 1st quartile, 3rd quartile and median values and outliers in field records.

sns.boxplot(user\_rating\_books['Age'])



***7.6.4 Remove Outliers:***

Find minimum and maximum threshold of data. The values above maximum threshold is given blow.

maximum\_threshold=user\_rating\_books['Age'].quantile(0.997)

maximum\_threshold

97.0

The maximum threshold is 97. To find total values above maximum threshold, use length function and apply sum function on x variable.

maximum\_values=np.where(user\_rating\_books['Age']>maximum\_threshold)

maximum\_values

(array([8291,8292, 8293, ..., 1021124, 1024670, 1026368],

dtype=int64),)

print(len(sum(maximum\_values)))

3032

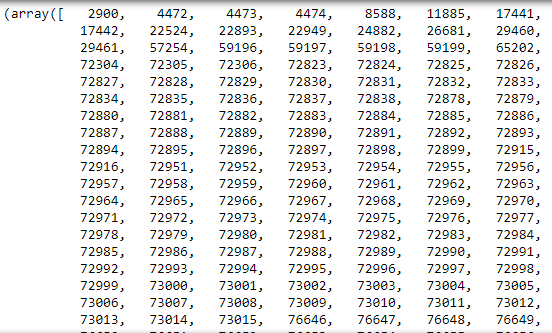
Find minimum threshold of “Age” column outlier. Use quantile function with percentage 0.001

And minimum threshold is 2.

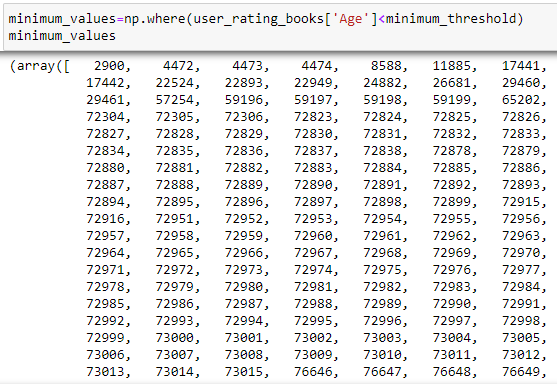
minimum\_threshold=user\_rating\_books['Age'].quantile(0.001)

minimum\_threshold

2.0



Total values less then minimum threshold found using where condition. Apply condition on “Age” column to display values less then minimum threshold.



Total values less then minimum threshold.

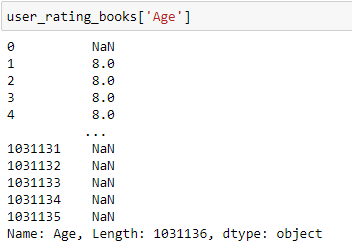
print(len(sum(minimum\_values)))

989

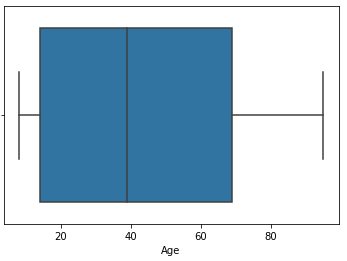
To remove outliers of “Age” column. Set condition where “Age” values greater then minimum threshold and minimum “Age” values then maximum threshold.

user\_rating\_books['Age']=user\_rating\_books[(user\_rating\_books['Age']>minimum\_threshold) & (user\_rating\_books['Age']<maximum\_threshold)]

user\_rating\_books['Age']



sns.boxplot(user\_rating\_books['Age'])



**7.7 Data binning/ Bucketing**

[Data bucketing](https://www.geeksforgeeks.org/python-binning-method-for-data-smoothing/) is another term for data binning. It's a data preparation technique that helps to reduce data observation errors. Values are divided into bins and replaced by general values produced for each bin in data binning.

In the case of a tiny data set, the method lowered the possibilities of overfitting.

***7.7.1 Importance of data binning***

* Preparation of machine learning models that required binned data.
* Provide protection against minor data errors.
* Provide protection against outliers.
* Vehicle for handling missing values (missing are assigned a special bin)
* Simplification, summarization and reporting.

The “Book-Rating” column was subjected to a data binning technique. Create a new column called "Age group" and fill it with binning values.

Create binning values in Python with the cut () function, and set the total number of bins required. The binning values should be saved in a column.

bins=[2,14,24,40,56,67,97]

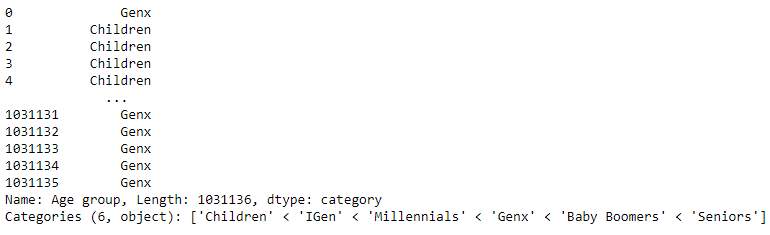
Make the binning of age values, from minimum threshold value 2-14 children, 15-24 years old users IGEN, 25-40 years old users Millennials, 40-56 years old users Genx users, 55-67 baby boomers and above 67-97 years old users are seniors.

group\_names=['Children','IGen','Millennials','Genx','Baby Boomers','Seniors']

user\_rating\_books['Age group']

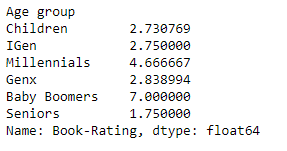
=pd.cut(user\_rating\_books['Age'],bins,labels=group\_names)

user\_rating\_books['Age group']



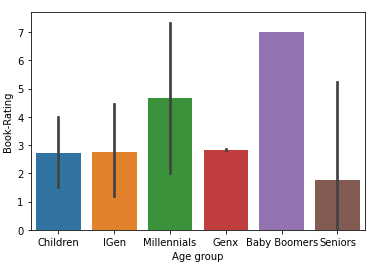
Find the average of “Book Rating” on binning. Use mean to find average. Use group by function and apply on average rate column. And find the mean value.

user\_rating\_books.groupby('Age group')['Book-Rating'].mean()



Display this binning field using bar plot graph. Apply “Age group” field on x-axis and book-rating on y-axis.

sns.barplot(user\_rating\_books['Age group'], user\_rating\_books['Book-Rating'])



**7.8 Understanding relationships and new insights using plots.**

We can get relationship of data by visualization. There are some techniques to visualize data. Visualization is a central part of data analysis for exploration and understanding of data.

* Histograms
* Heat maps
* Distribution plots
* Boxplots
* Bar plots

Visualization using Python Programming.

There are some Python libraries to visualize the data and relationships as “matplolib” and “seaborn”.

**7.8.1 Seaborn** is open source Python library built on the top of matplotlib and used for making statistical graphs. It provides beautiful style and color palettes and make more attractive graphs.

To import seaborn library use this method.

import seaborn as sns

**7.8.2 Matplotlib** is open source Python library works like MATPLOT. It works on Pyplot function which makes some change to figure, create figure, creates a plotting area in figure, plots some lines and decorate the plots with labels.

To import matplotlib library use this method.

import matplotlib.pyplot as plt

**7.9 Data visualization**

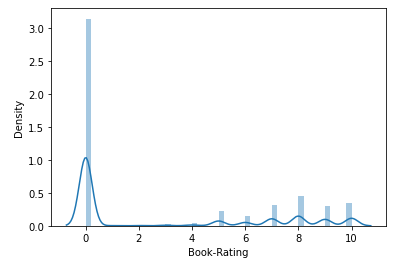
***7.9.1 Distribution******plot graph***

For numerical data, the distribution plot is useful for comparing range and distribution for groups. It's represented as a series of value points on a graph.

The “Book Rating” distribution of the users.

Data about the rating distribution of users. Check out the users' maximum and minimum rating ranges.

sns.distplot(user\_rating\_books['Book-Rating'])

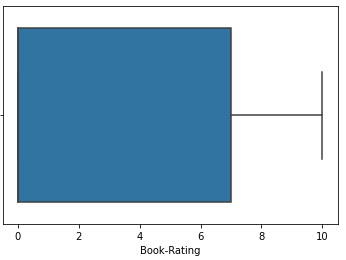


***7.9.2 Boxplot Visualization***

With a single box, a boxplot efficiently displays a summary of data. The 25th, 50th, and 75th percentiles are used to summarize data in a boxplot. Lower quartile, median, and higher quartile are other terms for them.

Use a box plot to visualize the “Book Rating” field data and locate quartiles and outliers. The boxplot 5 upper quartile. The lowest quartile is 3 and the median quartile is 4.

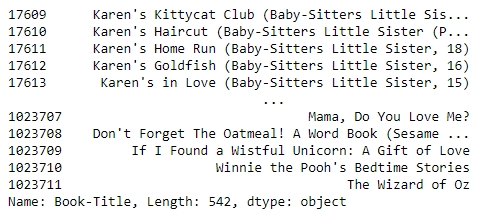
sns.boxplot(user\_rating\_books['Book-Rating'])



Find the books read by eight years old users. To find those books, set the condition on users “Age” field equal to 8.

result\_df=user\_rating\_books.loc[user\_rating\_books['Age']==8]

result\_df['Book-Title']



**7.9.3 Correlation**

Correlation coefficients are indicators of the strength of linear relationships between variables.

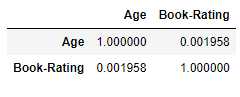
**Negative & Positive Correlation**

A linear correlation coefficient that is greater than zero signifies the positive relationship.

A value that is less than zero signifies a negative or inverse relationship.

A value of zero signifies no relationship between or among variables being compared.

user\_rating\_books[['Age’,’Book-Rating’]].corr()



**Heat Map**

It is a graphical representation of data, in which data values are represented in colors, delievering a convenient, insightful view of information.

It uses colors in order to communicate a value to the reader. This is greater tool to assist the audience towards the areas that matter the most when you have a larger value of data.

Heatmap is used to display correlation matrix. It has following steps.

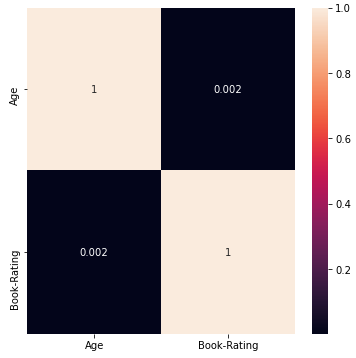
* Import data.
* Create correlation matrix.
* Creates heatmaps in seaborn.
* Export heatmaps.

To set the size of graph use matplotlib figure function and set the parameters. Use python seaborn library to display heatmap. Use correlation parameter to import correlation data.

On x-axis and y-axis set the correlation columns and make annotation true.

plt.figure(figsize=(7,7))

sns.heatmap(corelation, xticklabels=corelation.columns, yticklabels=corelation.columns, annot=True)



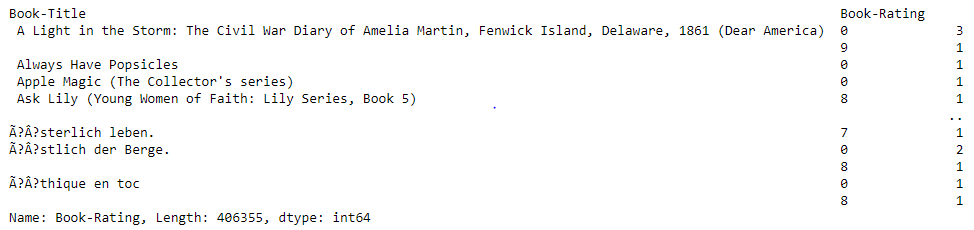
According to the values of correlation among variables. It signifies that there is no correlation

of variables because book value which is greater then 0 which is 1. And value of “Age” on x\_axis is less then 0.

**Check how many time books repeated and rated by users.**

Apply group by function on “Book Title” column, count books and number of ratings on each book.

user\_rating\_books.groupby('Book-Title')['Book-Rating'].value\_counts()



**7.9.4 Bar plot**

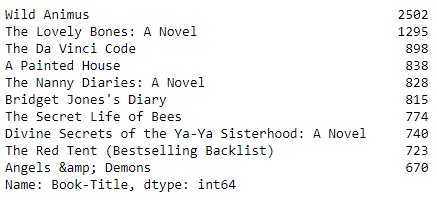
Bar plot is used to compare two different data elements. When try to measure change over time, bar graphs are best when measure changes are larger.

To check how many times a book is repeated in data set and read by different users. Find top 10 books which are mostly repeated in data. To find this count the values of books column.

Display the result using bar plot with sea born library set the key values of books in key named variable and values of books title in value named variable. Placed values variable on x-axis and key variable on y-axis.

Using value counts () method count the values of top 10 books.

user\_rating\_books['Book-Title'].value\_counts()[0:10]



Find the keys and values of “Book-Title” column and store the data in two different variables.

keys=list(user\_rating\_books['Book-Title'].value\_counts()

[0:10].keys())

values=list(user\_rating\_books['Book-Title'].value\_counts()

[0:10])

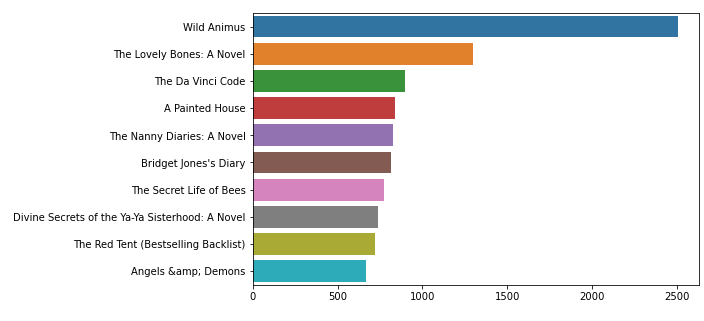
Set the figure size width 8 and height 5. Set the x-axis variable as values and y-axis variable as keys. display bar plot using sea born library.

plt.figure(figsize=(8,5))

sns.barplot(x=values, y=keys)

plt.show()

Display of bar plot graph



**7.9.5 Rating Count Distribution plot**

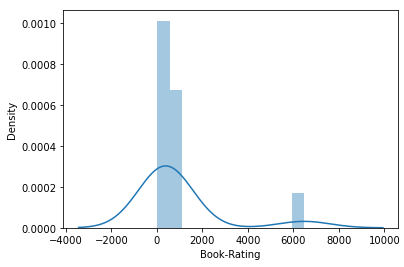
Find the maximum rating values of “Book-Rating” column. Apply value counts () function on field and store the data in total rating variable. Display the result using dist plot graph.

It displays above the range values. To control this divided the data of total rating variable by 100.

total\_rating=user\_rating\_books['Book-Rating'].value\_counts()

Divide the data stored in total rating variable to minimize the range of values for better visualization.

sns.distplot(total\_rating / 100 )



**Create random values of book category column with specific ratios.**

Create a column of book category with random values and set the values with specific ratios.

For random values import random library.

Create a list of books category. Set the different ratios of different types of books category.

Define random choices function and set the parameter with category, k defines the length of values and weight parameter defines ratios.

Make a data frame and set column name as DBookCategory.

import random

category=["History","Romance Novel","Fiction","Science Fiction","Horror","Fantasy","Nonfiction","Biography","Crime",

"Mystery","Poetry","Thriller","Adventure","Action"]

ratio=[0.12,0.7,0.8,0.10,0.7,0.4,0.5,0.5,0.6,0.6,0.10,0.5,0.6,0.9]

genure=random.choices(category, k=1031136, weights=ratio)

bookcategory=pd.DataFrame(genure, columns={'DBookCategory'})

**Create random values of gender column with specific ratios.**

Create a column of gender with random values and set the values with specific ratios.

For random values import random library.

Create a list of books gender values. Set the different ratios of different gender.

Define random choices function and set the parameter with gender, k defines the length of values and weight parameter defines ratios.

Make a data frame and set column name as Gender.

import random

gendr = ['Male', 'Female']

weight = [0.6, 0.4]

x=random.choices(tech, k=1031136, weights = weight)

*#for i in range(0, 79701):*

*#print(x)*

*#len(x)*

gender=pd.DataFrame(gendr, columns={'Gender'})

**Concatenate gender and category data frame with dataset.**

Pandas concatenate function is used to concatenate the data of user rating books data set with gender and book category.

user\_rating\_books=pd.concat([user\_rating\_books,gender,bookcategory], ignore\_index=False, axis=1)

**Count plot:**

count plot method is used to show the counts of observations in each categorical bin using bars.

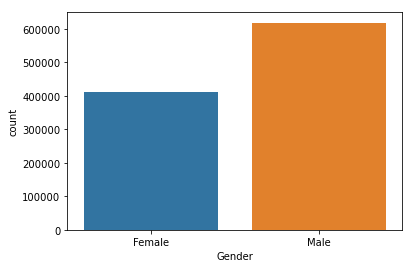
Count plot accepts two parameters to set axis. These parameters take name of variables in data or vector data, optional, inputs for plotting long form data.

**Creating a count plot of gender column.**

To count the values of male and female from gender colum apply countplot function.

Use seaborn library to apply countplot function. display the values of male is 60 and female is 40.

sns.countplot(user\_rating\_books["Gender"])



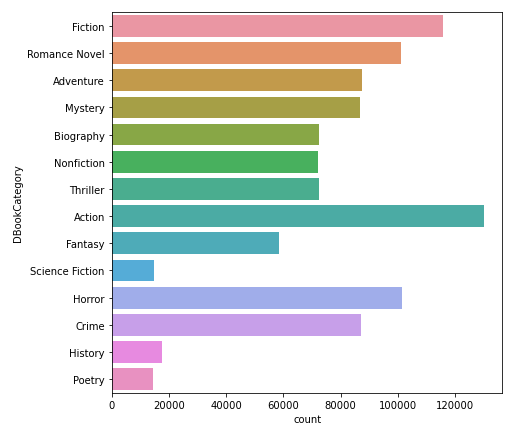
**Creating a count plot of DBookCategory.**

To count the values of BookCategory colum apply countplot function.

Use seaborn library to apply countplot function. display the values with different ratios.

plt.figure(figsize=(7,7))

sns.countplot(y=user\_rating\_books["DBookCategory"])



**Matrix Factorization**

Matrix factorization is also known as Matrix decomposition is class of collaborative filtering algorithms used for recommended systems. It decomposes the user item interaction matrix into product of two lower dimensionality rectangular metrics. Matrix factorization is mathematical model help the system split an entity into multiple smaller entries. Matrix factorization generates an output in the form of recommendations.

**Dimensionality Reduction**

Reducing the number of input variables for predictive model**.** The variables should be numerical data type.

**Singular Value Decomposition (SVD)**

Popular technique for dimensionality reduction in Machine Learning. SVD is used when data is sparse.

This is technique that comes from the field of linear algebra and can be used as data preparation technique to create a projection of sparse dataset prior to fitting a model.

SVD is also used in matrix operations, such as matrix inverse, but also as data reduction method in machine learning.

**Divide the data into train and test split**

Data is separated into training and testing parts in order to train the model. The training data ratio is usually higher than the testing data ratio. To assess the performance of a machine learning model, training data is used, and testing data is used to predict the outcome.

Import the train test split method from the sklearn machine learning library.

from sklearn.model\_selection import train\_test\_split

train\_df,test\_df=train\_test\_split(data,test\_size=.20, random\_state=40)

set the indexes of train and test data as ‘User-ID’ id of users.

interactions\_train\_indexed\_df = train\_df.set\_index('User-ID')

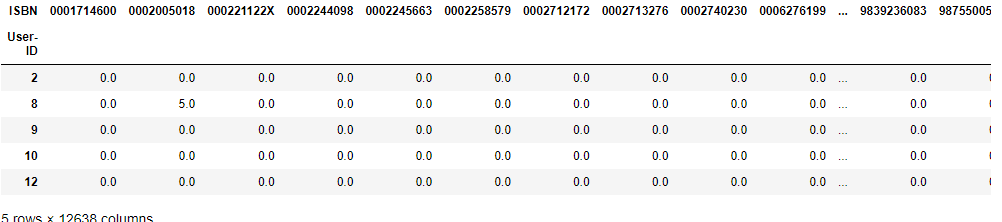
interactions\_test\_indexed\_df = test\_df.set\_index('User-ID')

**Pivot Table**

One of the most basic data analysis tools is the pivot table. Many key business questions can be easily answered with pivot tables. We develop Pivot Tables for a variety of reasons, one of which is to convey information. We'd like to back up our story with data that's simple to comprehend and see.

Set user-id as index values and columns ad ISBN numbers. ISBN numbers are uniquely identifying each book. And fill all missing values with zero.

pivot\_df=train\_df.pivot\_table(index='User-ID', columns='ISBN', values='Book-Rating').fillna(0)



create the matrix of pivot table data. To create matrix set values. To display the userid convert index values into list.

pivot\_matrix=pivot\_df.values

user\_ids=list(pivot\_df.index)

user\_ids

to convert matrix into sparse matrix use python scipy library and import csr\_matrix and fit pivot matrix.

from scipy.sparse import csr\_matrix

sparse\_matrix=csr\_matrix(pivot\_matrix)

import svd from scipy library from scipy linear algebra class. Fit the sparse matrix data and set k as number of components 12.

from scipy.sparse.linalg import svds

u,s,vt=svds(sparse\_matrix,k=12)

* create a digonal matrix of s (sigma) values

sigma = np.diag(s)

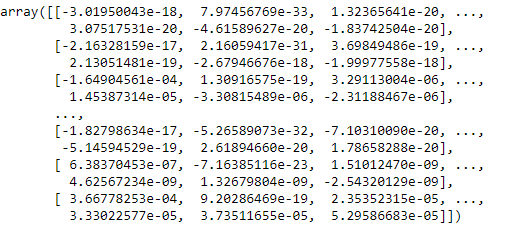
sigma.shape

(12, 12)

* create a dot product matrix of all three factors into single matrix.

all\_user\_predicted\_ratings=np.dot(np.dot(u,sigma),vt)

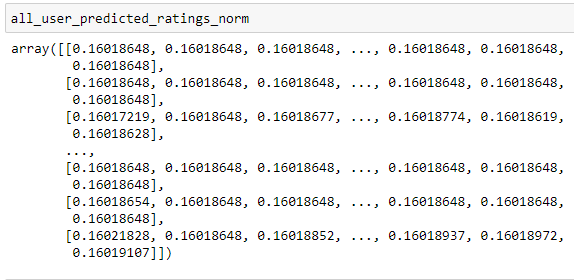
all\_user\_predicted\_ratings



* normalization of matrix to find average rating of users to each book.

all\_user\_predicted\_ratings\_norm=(all\_user\_predicted\_ratings-all\_user\_predicted\_ratings.min())/all\_user\_predicted\_ratings.max()-all\_user\_predicted\_ratings.min()

all\_user\_predicted\_ratings\_norm



* convert the matrix into a data frame.

cf\_pred\_df=pd.DataFrame(all\_user\_predicted\_ratings\_norm,columns=pivot\_df.columns, index=user\_ids).transpose()

cf\_pred\_df

* make a class name CFRecommender to recommend the specific books to users.

class CFRecommender:

def \_\_init\_\_(self, cf\_predictions\_df, items\_df=None):

self.cf\_predictions\_df = cf\_predictions\_df

self.items\_df = items\_df

def recommend\_items(self, user\_id, items\_to\_ignore=[], topn=10, verbose=False):

*Get and sort the user's predictions*

sorted\_user\_predictions = self.cf\_predictions\_df[user\_id].sort\_values(ascending=False) \

.reset\_index().rename(columns={user\_id: 'recStrength'})

* Recommend the highest predicted rating movies that the user hasn't seen yet.

recommendations\_df=sorted\_user\_predictions[~sorted\_user\_predictions['ISBN'].

isin(items\_to\_ignore)] \

.sort\_values('recStrength', ascending = False) \.head(topn)

if verbose:

if self.items\_df is None:

raise Exception('"items\_df" is required in verbose mode')

recommendations\_df = recommendations\_df.merge(self.items\_df, how ='left',

left\_on = 'ISBN',

right\_on='ISBN')

[['recStrength', 'ISBN']]

return recommendations\_df

cf\_recommender\_model = CFRecommender(cf\_pred\_df, articles\_df)

* To predict the result using test split data inspect intrections () function is called .

def inspect\_interactions(user\_id, test\_set=True):

if test\_set:

interactions\_df = interactions\_test\_indexed\_df

else:

interactions\_df = interactions\_train\_indexed\_df

return interactions\_df.loc[user\_id].merge(articles\_df, how = 'left',

left\_on = 'ISBN',

right\_on = 'ISBN')

inspect\_interactions(7552, test\_set=False).head(20)

