```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df=pd.read_csv('books.csv', error_bad_lines=False)
df=df.drop(['Unnamed: 12'], axis=1)
df.head()
df.isnull().sum()
df.info()
df.describe()
top_ten=df[df['ratings_count']>1000000]
top_ten=top_ten.sort_values(by='ratings_count', ascending=False).head(10)
top_ten_rating=top_ten.sort_values(by='average_rating', ascending=False)

plt.style.use('seaborn-whitegrid')
plt.figure(figsize=(20,20))
sns.barplot(x="average_rating", y="title", data=top_ten_rating, palette='inferno')
book_aut=df.groupby('authors')['title'].count().reset_index().sort_values('title',
ascending=False).head(10).set_index('authors')
book_aut
plt.figure(figsize=(20,15))
ax=sns.barplot(book_aut['title'], book_aut.index, palette='inferno')
ax.set_title("Top 10 authors  with most books")
ax.set_xlabel("Total number of books")

total=[]

for i in ax.patches:A
    total.append(i.get_width())

totals=sum(total)

for i in ax.patches:
    ax.text(i.get_width()+.2, i.get_y()+.2, str(round(i.get_width())))

plt.show()
plt.figure(figsize=(20,15))

ax=sns.barplot(most_rated['ratings_count'], most_rated.index, palette='inferno')
ax.set_title("Top 10 books  with most rated")


total=[]
```

```
for i in ax.patches:
    total.append(i.get_width())

totals=sum(total)

for i in ax.patches:
    ax.text(i.get_width()+.2, i.get_y()+.2, str(round(i.get_width())))

plt.show()
df=df.sort_values('average_rating').reset_index()
df=df.iloc[4:]
df.average_rating=df.average_rating.astype(float)
fig, ax= plt.subplots(figsize=[15,10])

sns.distplot(df['average_rating'], ax=ax)
ax.set_title('Average rating distribution for all books')
ax.set_xlabel('average rating')
plt.figure(figsize=(20,20))
ax=sns.relplot(data=df, x="average_rating", y="ratings_count",sizes=(400,200), height=7)
plt.title('Relation between Rating counts and avg rating')
ax.set_axis_labels("average_rating", "ratings_count")
df=df.sort_values('num_pages', ascending=False).reset_index()
df=df.iloc[4:]
df.head(20)
df.num_pages=df.num_pages.astype(float)
plt.figure(figsize=(20,20))
ax=sns.relplot( x="average_rating", y="num_pages",data=df,sizes=(400,200), height=7)
plt.title('Relation between Rating counts and no pages')
ax.set_axis_labels("average_rating", "no pages")
df2=df.copy()
df2.loc[(df2['average_rating']>=0)&(df2['average_rating']<=1),'rating between']="between 0 and 1"
df2.loc[(df2['average_rating']>1)&(df2['average_rating']<=2),'rating between']="between 1 and 2"
df2.loc[(df2['average_rating']>2)&(df2['average_rating']<=3),'rating between']="between 2 and 3"
df2.loc[(df2['average_rating']>3)&(df2['average_rating']<=4),'rating between']="between 3 and 4"
df2.loc[(df2['average_rating']>4)&(df2['average_rating']<=5),'rating between']="between 4 and 5"
df2.sort_values('average_rating')
rating_df=pd.get_dummies(df2['rating between'])
language_df=pd.get_dummies(df2['language_code'])
feature=pd.concat([rating_df, language_df, df2['average_rating'], df2['ratings_count']], axis=1)
from sklearn.preprocessing import MinMaxScaler
minmax=MinMaxScaler()
feature=minmax.fit_transform(feature)
from sklearn import neighbors
from sklearn.model_selection import train_test_split
```

```python
model=neighbors.NearestNeighbors(n_neighbors=8, algorithm='ball_tree')

model.fit(feature)

dist, idlist=model.kneighbors(feature)
def bookRecom(name):
    book_list=[]
    book_id=df2[df2['title']==name].index
    book_id=book_id[0]
    for new in idlist[book_id]:
        book_list.append(df2.loc[new].title)
    return book_list

Book_name=bookRecom('Harry Potter and the Half-Blood Prince (Harry Potter  #6)')
Book_name
```