MIT Academy of Engineering
(An Autonomous Institute Affiliated to Savitribai Phule University)

THEORY ACTIVITY NO. 1

ESSENTIALS OF DATA SCIENCE

DATASET: GOODREADS BOOK REVIEWS

BY CS3-71 CHETAN BALIRAM GORE

PRN:-202401040350

MIT-AOE 2024-45

1. FIND THE TOTAL NUMBER OF BOOKS AND BASIC INFORMATION:-

```
import pandas as pd
import numpy as np

# Load the dataset
df = pd.read_csv('goodreads_books.csv')

# Display shape
print(f"Total number of books: {df.shape[0]}")
print(f"Total number of features (columns): {df.shape[1]}")

# Display basic info
print("\nDataset Information:")
print(df.info())

# Display first few rows
print("\nSample Data:")
print(df.head())
```

2. CALCULATE THE OVERALL AVERAGE RATING OF ALL BOOKS AND CHECK DISTRIBUTION:-

```
# Average rating
average_rating = df['rating'].mean()
print(f"\nAverage Rating of all books:
{average_rating:.2f}")

# Rating distribution
print("\nRating Value Counts:")
print(df['rating'].value_counts().sort_index())
```

3. FIND THE BOOK WITH THE HIGHEST RATING ALONG WITH FULL DETAILS:-

```python
# Highest rated book
highest_rating = df['rating'].max()
top_books = df[df['rating'] == highest_rating]

print(f"\nBooks with the highest rating ({highest_rating}):")
print(top_books[['title', 'author', 'rating', 'number_of_reviews']])
```

4. FIND THE BOOK WITH THE LOWEST RATING:-

```
# Lowest rated book
lowest_rating = df['rating'].min()
low_books = df[df['rating'] == lowest_rating]

print(f"\nBooks with the lowest rating ({lowest_rating}):")
print(low_books[['title', 'author', 'rating', 'number_of_reviews']])
```

5. LIST TOP 10 BOOKS WITH MAXIMUM REVIEWS:-

```
# Sorting based on number of reviews
top10_reviews = df.sort_values(by='number_of_reviews', ascending=False).head(10)

print("\nTop 10 books with the highest number of reviews:")
for i, row in top10_reviews.iterrows():
    print(f"{row['title']} by {row['author']} - {row['number_of_reviews']} reviews")
```

6. FIND THE MOST FREQUENT AUTHOR AND HOW MANY BOOKS THEY HAVE:-

```
# Author with most books
most_frequent_author = df['author'].mode()[0]
author_count = df['author'].value_counts()[most_frequent_author]

print(f"\nMost frequent author: {most_frequent_author} with {author_count} books.")
```

7. CALCULATE THE AVERAGE NUMBER OF PAGES ACROSS ALL BOOKS:-

```
# Average pages
avg_pages = df['pages'].mean()
median_pages = df['pages'].median()

print(f"\nAverage number of pages: {avg_pages:.2f}")
print(f"Median number of pages: {median_pages}")
```

8. IDENTIFY BOOK WITH MAXIMUM PAGES:-

```
# Book with max pages
max_pages = df['pages'].max()
max_page_book = df[df['pages'] == max_pages]

print(f"\nBook with maximum pages ({max_pages} pages):")
print(max_page_book[['title', 'author', 'pages']])
```

9. FIND BOOKS WITH A RATING GREATER THAN 4.5:-

```
# Books with rating > 4.5
excellent_books = df[df['rating'] > 4.5]

print(f"\nTotal books with rating > 4.5: {excellent_books.shape[0]}")
print(excellent_books[['title', 'author', 'rating']].head(10))  # Display sample
```

10. NUMBER OF BOOKS PUBLISHED EACH YEAR:-

```
# Books per publication year
books_by_year = df['publication_year'].value_counts().sort_index()

print("\nNumber of books published each year:")
print(books_by_year)
```

11. FIND THE AVERAGE RATING PER AUTHOR AND LIST TOP 5:-

```
# Average rating per author
author_avg_rating = df.groupby('author')['rating'].mean().sort_values(ascending=False)

print("\nTop 5 Authors by Average Rating:")
print(author_avg_rating.head(5))
```

12. MOST COMMON GENRE:-

```
# Most common genre
most_common_genre = df['genre'].mode()[0]

print(f"\nMost common genre among all books: {most_common_genre}")
```

13. CORRELATION BETWEEN NUMBER OF PAGES AND RATING:-

```
# Correlation
correlation_value = df['pages'].corr(df['rating'])

print(f"\nCorrelation between number of pages and rating: {correlation_value:.2f}")
```

14. TOTAL NUMBER OF UNIQUE AUTHORS:-

```
# Unique authors
unique_authors_count = df['author'].nunique()

print(f"\nTotal number of unique authors: {unique_authors_count}")
```

15. REPLACE MISSING VALUES PROPERLY:-

```
# Fill missing values
df['rating'].fillna(df['rating'].mean(), inplace=True)
df['pages'].fillna(df['pages'].median(), inplace=True)
df['genre'].fillna('Unknown', inplace=True)

print("\nMissing values handled successfully.")
```

16. LIST BOOKS WITH TITLE LENGTH > 50 CHARACTERS:-

```
# Books with long titles
long_title_books = df[df['title'].apply(lambda x: len(x) > 50)]

print(f"\nBooks with title length greater than 50 characters:")
print(long_title_books[['title', 'author']].head(10))
```

17. IDENTIFY BOOKS WITH ZERO REVIEWS:-

```
# Books with 0 reviews
zero_reviews = df[df['number_of_reviews'] == 0]

print(f"\nTotal books with zero reviews: {zero_reviews.shape[0]}")
print(zero_reviews[['title', 'author']])
```

18. CREATE PIVOT TABLE: AVERAGE RATING PER GENRE AND PUBLICATION YEAR:-

```
# Pivot Table
pivot_avg_rating = pd.pivot_table(df, values='rating', index='genre', columns='publication_year',
aggfunc=np.mean)

print("\nPivot Table: Average Rating per Genre and Year")
print(pivot_avg_rating)
```

19. NORMALIZE NUMBER OF REVIEWS:-

```
# Min-Max Normalization
df['normalized_reviews'] = (df['number_of_reviews'] - df['number_of_reviews'].min()) /
(df['number_of_reviews'].max() - df['number_of_reviews'].min())

print("\nNormalized 'number_of_reviews' column added successfully.")
print(df[['title', 'normalized_reviews']].head(10))
```

20. FIND TOP-RATED AUTHOR WHO HAS WRITTEN MINIMUM 5 BOOKS:-

```
# Author with minimum 5 books
authors_with_5_books = df['author'].value_counts()
valid_authors = authors_with_5_books[authors_with_5_books >= 5].index

# Calculate average ratings
top_authors = df[df['author'].isin(valid_authors)].groupby('author')
['rating'].mean().sort_values(ascending=False)

print("\nTop-rated author with at least 5 books:")
print(top_authors.head(1))
```

MIT-AOE 2024-45

THANK YOU!

CS3-71 CHETAN BALIRAM GORE