MAHATMA EDUCATION SOCIETY'S

PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE
(Autonomous)

NEW PANVEL

ANALYTIC PROJECT ON

"College Library Data Usage"

IN PARTIAL FULFILLMENT OF

BACHELOR OF COMPUTER SCIENCE

SEMESTER IV – 2025-2026

PROJECT GUIDE

Name : **Mrs. Sanjana Bhangale**

SUBMITTED BY : Dhanshree Agiwale

ROLL NO : **5257**

Class: BSC CS (SY)

Subject : Data Science

Details about the continuous Assessment 2/Project work

Name of the Student : **Dhanshree Agiwale**

Roll Number : **5257** Class / Division : SYCS (A)

Name of Evaluator: Prof. Sanjana Bhangale

Please circle appropriate score

Grading Criteria	Fair	Good	Excellent	Total
Introduction/ Description of the Case	1	2	3	/3
SWOT Analysis of the company used for case analysis pertaining to the case (**strength** of CA2 topic: e.g main important feature of CA1, **Weakness:** limitations of the project, **Opportunities:** in carrier in the future, **Threat:** obstacles that can cause failure to project CA2	3	4	5	/5
Learnings from the case	2	3	4	/4
Delivery/presentation skills	1	2	3	/3
Total				/15

Co-ordinator,

Project Title:

College Library Data Usage Analysis

Introduction

Libraries play an important role in students' academic life. They support learning by providing access to books, study space, and resources.
This project focuses on analyzing **college library usage data** using **Python and Pandas** to understand student behavior and library patterns.

The dataset contains information such as student details, department, year of study, book genres, borrowing habits, study hours, visit frequency, and fines paid.
By performing structured data analysis, meaningful insights are extracted that help understand how students use the library.

Objectives of the Project

The main objectives of this project are:

- To analyze **student library usage patterns**
- To understand **borrowing behavior** across departments and years
- To study the relationship between **study hours, visit frequency, and books borrowed**
- To identify **fine patterns and overdue behavior**
- To classify students based on **library usage level**
- To perform **basic, intermediate, and advanced Pandas operations**

Tools & Technologies Used

- **Python**
- **Pandas** – data cleaning and analysis
- **Jupyter Notebook** – interactive analysis environment

Dataset Description

The dataset includes the following types of information:

- Student name, department, and year of study
- Library membership type (Standard / Premium)
- Book genre preference
- Number of books borrowed
- Days books were kept

- Study hours per week
- Library visit frequency
- Borrow status (Returned / Overdue / None)
- Fine amount paid

This data is cleaned, transformed, and analyzed to answer **50 structured analysis questions**.

Analysis Levels Covered

☐ **Basic Analysis (1–15)**

- Data loading and inspection
- Data cleaning and formatting
- Counting students by department, year, genre
- Extracting numeric values
- Identifying fined and frequent users

☐ **Intermediate Analysis (16–30)**

- Grouping and aggregation
- Comparing memberships
- Correlation analysis
- Sorting and filtering
- Pivot tables and monthly trends

☐ **Advanced Analysis (31–50)**

- Text extraction
- Feature engineering
- Usage classification
- Outlier detection
- Ranking and rule-based prediction

Outcome of the Project

This project demonstrates how **real-world educational data** can be analyzed using Pandas to gain insights into student behavior.
It improves understanding of:

- Library usage trends
- Student study habits
- Fine and overdue risks
- Data cleaning and analysis techniques

Data Analysis Question:

Basic Pandas Analysis Questions (1–15):

```
#1. Load the dataset into a pandas DataFrame and display the first and last 5 rows.
import pandas as pd

df = pd.read_csv(
    r"C:\Users\dell\Downloads\collage library data track .xlsx - collage_librar (2).csv",
    header=None,
    engine="python"
)
df = df.iloc[2:].reset_index(drop=True)
df.columns = [
    "DATE of joining library",
    "Name of Student",
    "Department",
    "Year of study",
    "Book Genre",
    "Library Membership Type",
    "Borrow Status",
    "Books Borrowed Per Month",
    "Days Kept Books",
    "Fine Paid (₹)",
    "Study Hours Per Week",
    "Library Visits Per Week",
    "Extra1", "Extra2", "Extra3", "Extra4", "Extra5"
]
df = df.drop(columns=["Extra1", "Extra2", "Extra3", "Extra4", "Extra5"])
df.head()
```

	DATE of joining library	Name of Student	Department	Year of study	Book Genre	Library Membership Type	Borrow Status	Books Borrowed Per Month	Days Kept Books	Fine Paid (₹)	Study Hours Per Week	Library Visits Per Week
0	DATE of joing library	Name of Student (Full Name)	Department	Year of study	Book Genre You Usually Read\n	Library Membership Type\n	Borrow Status	Number of Books you Borrowed in a month (e.g. ...	How Many Days You Kept the Books ?\n	Fine Paid (₹)\n	Your Average Study Hours in Library Per Week\n	How many times you visit library in a week?
1	1/4/2025	Yash surve	Bsc cs	SY	Academic / Educational	Standard	Returned	1–2	4–7 days	0	3–5 hours	2–3 times
2	10/10/2025	sanika gole	bsc cs	SY	Self-help / Motivational	Premium	Overdue	1–3	8–14 days	100	3–5 hours	2–3 times
3	1/5/2025	saniya koli	bsc it	TY	Historical	Standard	NaN	1–4	0–3 days	0	0–2 hours	4–5 times
4	12/10/2025	Kalpana irle	bca	FY	Historical	Standard	Overdue	1–5	0–3 days	150	9+ hours	6+ times

Analytical Statement: Dataset was loaded into Python suressfully. It contains student library details.

```
#2. Convert DATE of joining library column into proper datetime format.
df["DATE of joining library"] = pd.to_datetime(
    df["DATE of joining library"],
    errors="coerce"
)

df.dtypes
```

```
C:\Users\dell\AppData\Local\Temp\ipykernel_23208\3395237024.py:1: UserWarning: Could
ng back to `dateutil`. To ensure parsing is consistent and as-expected, please specif
  df["DATE of joining library"] = pd.to_datetime(
DATE of joining library     datetime64[ns]
Name of Student                    object
Department                         object
Year of study                      object
Book Genre                         object
Library Membership Type            object
Borrow Status                      object
Books Borrowed Per Month           object
Days Kept Books                    object
Fine Paid (₹)                      object
Study Hours Per Week               object
Library Visits Per Week            object
dtype: object
```

Analytical Statement: The dataset shows dept and borrowing info. Study hours and visits are recorded

```
#3. Count how many students are from each Department.
df["Department"].value_counts()
```

```
Department
Bcom          12
Bms           10
Baf            8
Bsc it         8
Bsc cs         7
Bca            6
Commerce       4
Bsc it         4
Bsc            4
```

Analytical Statement: missing values were identified. Incorrect entries were corrected. Clean data improves accuracy

```
#4. Find the total number of students in the dataset.
df.shape[0]
```

```
110
```

Analytical Statement: Some dept use library more. borrowing varies across dept. shows usage different

```
#5. Count how many students belong to each Year of Study (FY, SY, TY).
df["year_of_study"].value_counts()
```

```
year_of_study
SY              41
TY              32
FY              25
Other            2
Year of study    1
Name: count, dtype: int64
```

Analytical Statement: Library visit differ by year senior students visit more often. indicates study needs

```
#6. Display all data for students who have Premium Membership.
df[df["library_membership_type"] == "Premium"]
```

	date_of_joining_library	name_of_student	department	year_of_study	book_genre	library_membership_type	borrow_status	books_borrowed_per_month	days
2	2025-10-10	sanika gole	bsc cs	SY	Self-help / Motivational	Premium	Overdue	1–3	
6	2025-03-28	Omkar shinde	Bsc it	SY	Fiction	Premium	Returned	3–4	
7	2025-01-19	Rutika pawar	Bsc it	SY	Academic / Educational	Premium	Overdue	3–4	
8	2025-07-15	Sonali yewale	Baf	SY	Academic / Educational	Premium	Overdue	5–6	
10	2025-11-30	Sushant lohar	Bca	FY	Academic / Educational	Premium	Overdue	5–6	
11	2025-08-16	sakshi mhatre	bca	SY	Self-help / Motivational	Premium	Overdue	5–6	

Analytical Statement: Academic books are most borrowed. Genre preference is visible

```
#7. Count how many students returned the books vs overdue vs none.
df["borrow_status"].value_counts()
```

```
borrow_status
Overdue          53
Returned         28
Borrow Status     1
Name: count, dtype: int64
```

Analytical Statement: Some students borrow frequently. others borrow occasionally. shows usage variation Overdue = 53, Return → 28, Brow → 1.

```
#8. Identify how many days books were kept by extracting numerical values from How Many Days You Kept the Books?
df["days_kept_books"].head(10)
```

```
0    How Many Days You Kept the Books ?\n
1                                  4-7 days
2                                 8-14 days
3                                  0-3 days
4                                  0-3 days
5                                  4-7 days
6                                  15+ days
7                                  4-7 days
8                                 8-14 days
9                                  0-3 days
Name: days_kept_books, dtype: object
```

Analytical Statement: Study hours vary among students. Regular visitors & study longer. Indicates discipline

```
#9. Standardize the column names (snake_case).
df.columns = [col.strip().lower().replace(' ', '_').replace('/', '_') for col in df.columns]
df.columns
```

```
Index(['date_of_joining_library', 'name_of_student', 'department',
       'year_of_study', 'book_genre', 'library_membership_type',
       'borrow_status', 'books_borrowed_per_month', 'days_kept_books',
       'fine_paid_rs', 'study_hours_per_week', 'library_visits_per_week',
       'date_of_joining_library', 'days_kept_numeric'],
      dtype='object')
```

Analytical Statement: Premium members are more active. they brow more book. membership impacts usage.

```
#10. Extract only numbers from Number of Books Borrowed column.
df['books_borrowed_per_month'] = df['books_borrowed_per_month'].astype(str).str.replace(r'[^\d\--]', '', regex=True)
df['books_borrowed_per_month'].head(10)
```

```
0    356
1    1-2
2    1-3
3    1-4
4    1-5
5    1-6
6    3-4
7    3-4
8    5-6
9    3-4
Name: books_borrowed_per_month, dtype: object
```

Analytical Statement: Fine records show Overdue behavior. late returns increas fines. helps identify patterns.

```
#11. Identify how many days books were kept by extracting numerical values from How Many Days You Kept the Books?

import re
import numpy as np

def replace_range(val):
    if pd.isna(val) or val == '':
        return np.nan
    if '-' in val or '–' in val:
        numbers = re.findall(r'\d+', val)
        numbers = [int(n) for n in numbers]
        return sum(numbers) / len(numbers)
    return float(val)

df['books_borrowed_per_month_numeric'] = df['books_borrowed_per_month'].apply(replace_range)

# Check the first 10 rows
df[['books_borrowed_per_month', 'books_borrowed_per_month_numeric']].head(10)
```

	books_borrowed_per_month	books_borrowed_per_month_numeric
0	356	356.0
1	1–2	1.5
2	1–3	2.0
3	1–4	2.5

Analytical Statement:

frequent visitors borrow more. library supports learning.

[93]:
```
#12. List students who visit the library more than 4 times per week.
df['library_visits_numeric'] = df['library_visits_per_week'].astype(str).str.extract(r'(\d+)').astype(float)
frequent_visitors = df[df['library_visits_numeric'] > 4]
frequent_visitors
```

[93]:

	date_of_joining_library	name_of_student	department	year_of_study	book_genre	library_membership_type	borrow_status	books_borrowed_per_month	days_kept
4	2025-12-10	Kalpana irle	bca	FY	Historical	Standard	Overdue	1–5	0
13	2025-06-26	Xyz	Abc	Other	Other	Premium	NaN	7	1
27	2025-03-31	Bhushan kumar	Bsc cs	FY	Historical	Standard	Returned	5–6	4
42	2025-11-01	Shivani	Manegement	Other	Self-help / Motivational	Standard	Returned	7	8-
50	2025-08-14	Gaurang patil	Bcom	FY	Fiction	Premium	Overdue	5–6	4
51	2025-06-15	Perna naik	Baf	SY	Historical	Premium	Returned	5–6	4

Analytical Statement: Data formatting was improved. values standardized. easier to analyze.

```
#13. Display all students who read Self-help / Motivational genre.
self_help_students = df[df['book_genre'].str.lower().str.contains('self-help|motivational', na=False)]
self_help_students
```

	date_of_joining_library	name_of_student	department	year_of_study	book_genre	library_membership_type	borrow_status	books_borrowed_per_month	days_kept_t
2	2025-10-10	sanika gole	bsc cs	SY	Self-help / Motivational	Premium	Overdue	1–3	8-14
5	2025-12-28	Tushar bhosale	Baf	SY	Self-help / Motivational	Standard	Overdue	1–6	4-7
11	2025-08-16	sakshi mhatre	bca	SY	Self-help / Motivational	Premium	Overdue	5–6	4-7

Analytical Statement: numeric values extracted correctly supports calculations. improves analysis.

[97]:
```
#14. Check for missing or null values.
missing_values = df.isnull().sum()
print("Missing/null values per column:\n", missing_values)
```

Analytical Statement:

```
Missing/null values per column:
 date_of_joining_library          1
name_of_student                   9
department                        9
year_of_study                     9
book_genre                        9
library_membership_type           9
borrow_status                    28
books_borrowed_per_month          0
```

Brrow status track responsibility. Overdue cases are visible and encourage timely returns.

```
#15. Find all students who paid any fine > 0.

df['fine_paid_rs'] = df['fine_paid_rs'].astype(str).str.replace(r'[^\d]', '', regex=True)
# Step 2: Convert to numeric
df['fine_paid_rs'] = pd.to_numeric(df['fine_paid_rs'], errors='coerce')
# Step 3: Filter students who paid any fine > 0
students_with_fine = df[df['fine_paid_rs'] > 0]
students_with_fine
```

	date_of_joining_library	name_of_student	department	year_of_study	book_genre	library_membership_type	borrow_status	books_borrowed_per_month	days_kept_
2	2025-10-10	sanika gole	bsc cs	SY	Self-help / Motivational	Premium	Overdue	1–3	8-1
4	2025-12-10	Kalpana irle	bca	FY	Historical	Standard	Overdue	1–5	0
5	2025-12-28	Tushar bhosale	Baf	SY	Self-help / Motivational	Standard	Overdue	1–6	4
7	2025-01-19	Rutika pawar	Bsc it	SY	Academic / Educational	Premium	Overdue	3–4	4
8	2025-07-15	Sonali yewale	Baf	SY	Academic / Educational	Premium	Overdue	5–6	8-1

Analytical Statement: Counting reveals usage trends. shows Student activity levels. helps comparision

Intermediate Pandas Analysis Questions (16–30):

[35]:
```
#16. Group by Department and calculate the average number of books borrowed.
df.groupby("department").size()
```

[35]:
```
department
 psychology          1
Abc                  1
Architecture         1
Arts                 2
Ba                   1
Baf                  8
Bca                  6
Bcom                12
Bcs cs               1
Bcs it               1
```

Analytical Statement: Grouping show dept patterns. Highlights borrowing differences. useful comparison.

[204]:
```
df["books_borrowed_numeric"] = pd.to_numeric(
    df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"],
    errors="coerce"
)
df.groupby("department")["books_borrowed_numeric"].mean()
```

Analytical Statement: Aggregation cal (Mean) averages.

[204]:
```
department
 psychology          NaN
Abc                  NaN
Architecture         NaN
Arts                 NaN
Ba                   NaN
Baf                  0.0
```

Study hours measured. show trends
BAF → 0.0
rest of → NaN

[202]:
```
#17. Group by Book Genre and find the total fine collected.
df["fine_paid_numeric"] = pd.to_numeric(df["fine_paid_₹"], errors="coerce")
df.groupby("book_genre_you_usually_read")["fine_paid_numeric"].sum()
```

[202]:
```
book_genre_you_usually_read
Academic / Educational                                                          1150.0
Academic / Educational, Biography / Autobiography                                250.0
Academic / Educational, Fiction, Self-help / Motivational, Biography / Autobiography    100.0
Academic / Educational, Historical                                               200.0
Academic / Educational, Self-help / Motivational                                 300.0
Biography / Autobiography                                                       1150.0
Fiction                                                                         1650.0
Fiction, Biography / Autobiography, Historical                                   100.0
Fiction, Historical                                                              150.0
Historical                                                                      2120.0
Other                                                                            350.0
Self-help / Motivational                                                        2550.0
Self-help / Motivational, Biography / Autobiography                              100.0
Name: fine_paid_numeric, dtype: float64
```

Analytical Statement: fined paid by avarage of book category of each ~~st~~ ~~600~~ Stream highest 1150, lowest 100.

```
#18. Find the maximum and minimum number of study hours per week.
df["study_hours_numeric"].agg(["min","max"])
```

```
min    0.0
max    9.0
Name: study_hours_numeric, dtype: float64
```

Analytical Statement: min → 0.0
max → 9
Study hours by students per weeks.

```
[48]: #19. Compare: Students with Standard vs Premium library membership - who borrowed more books?
      df.groupby("library_membership_type")["books_borrowed_numeric"].mean()
```

Analytical Statement:

```
[48]: library_membership_type
      Premium     3.176471
      Standard    2.571429
      Name: books_borrowed_numeric, dtype: float64
```

membership comparison done. Premium vs standard usage seen.

```
#20. Calculate the range of days kept for each borrow status (Returned, Overdue, None).
df.groupby("borrow_status")["days_kept_numeric"].apply(lambda x: x.max() - x.min())
```

```
borrow_status
Overdue     15.0
Returned    15.0
Name: days_kept_numeric, dtype: float64
```

Analytical Statement: borrow status of book range in days
Overdue → 15
Return → 15

```
#21. Group by Year of Study and find the average fine paid.
df["fine_paid_numeric"] = pd.to_numeric(df["fine_paid__₹"], errors="coerce")
df.groupby("year_of_study")["fine_paid_numeric"].mean()
```

```
year_of_study
FY         90.000000
Other     150.000000
SY        109.756098
TY        102.187500
Name: fine_paid_numeric, dtype: float64
```

Analytical Statement: Fine paid by group of year ~~of~~ Student like fy, SY, TY and others
↓ ↓ ↓ ↓
90 150 102 109

```
#22. Determine which genre is most popular among FY/SY/TY students.
df.groupby(["year_of_study","book_genre_you_usually_read"]).size().sort_values(ascending=False)
```

```
year_of_study  book_genre_you_usually_read
SY             Academic / Educational                10
               Self-help / Motivational               9
TY             Fiction                                8
SY             Historical                             7
TY             Self-help / Motivational               7
               Historical                             7
               Biography / Autobiography              6
FY             Historical                             6
```

Analytical Statement: most popular genre students from Stream SY, TY, SY, TY.

```
#23. Sort students by fine paid in descending order.
df["fine_paid_numeric"] = pd.to_numeric(df["fine_paid__₹"], errors="coerce")
df_sorted = df.sort_values("fine_paid_numeric", ascending=False)[
    ["name_of_student_full_name", "fine_paid_numeric"]
]
```

Analytical Statement: Sorted student according by fine paid in decesing Order
om → 250
Omkar → 250

```
df_sorted.head()
```

	name_of_student_full_name	fine_paid_numeric
16	om pawar	250.0
81	Omkar shinde	250.0
68	Krunal karulkar	250.0
57	Sharmila Tagore	250.0
88	Erica	250.0

```
#24. Find the number of students who kept books for more than 7 days.
df[df["days_kept_numeric"] > 7].shape[0]
```

```
34
```

Analytical Statement: Student who keep books from collage library for more than 7 days are 34 students.

```
#25. Extract numeric values from study hours per week and convert to integer.
df["study_hours_numeric"]
```

Analytical Statement:

```
0      3.0
1      3.0
2      0.0
3      9.0
4      3.0
      ...
104    NaN
105    NaN
106    NaN
107    NaN
108    NaN
Name: study_hours_numeric, Length: 109, dtype: float64
```

```
#26. Compare visiting frequency between Standard and Premium members.
df.groupby("library_membership_type")["library_visits_numeric"].mean()
```

```
library_membership_type
Premium     2.784314
Standard    2.285714
Name: library_visits_numeric, dtype: float64
```

Analytical Statement: Preminum member visits library more frequently than standards. this shows higher engagment among Premium users.

```
#27. Identify students who borrowed 5 or more books in a month.
df[df["books_borrowed_numeric"] >= 5][["name_of_student_full_name","books_borrowed_numeric"]]
```

	name_of_student_full_name	books_borrowed_numeric
7	Sonali yewale	5.0
9	Sushant lohar	5.0
10	sakshi mhatre	5.0
11	Vedika sonawane	5.0

Analytical Statement: A few students borrowed 5 or more books in a month. these students are active libray user.

```
#28. Calculate correlation between:
• fine paid
• number of books borrowed
• days kept
import pandas as pd
import numpy as np
df["fine_paid_numeric"] = pd.to_numeric(df["fine_paid_₹"], errors="coerce")

def range_to_median(x):
    if pd.isna(x):
        return np.nan
    x = str(x).replace('–','-').replace(' ','')  # handle en-dash and spaces
    parts = x.split('-')
    nums = [float(p) for p in parts if p.isdigit()]
    return np.median(nums) if nums else np.nan

df["books_borrowed_numeric"] = df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"].apply(range_to_median)
df["days_kept_numeric"] = df["how_many_days_you_kept_the_books_?"].apply(range_to_median)

df[["fine_paid_numeric","books_borrowed_numeric","days_kept_numeric"]].corr()
```

Analytical Statement:

fin amount ↑es when book↓ ar↓ kept for mor↓ day↓.

	fine_paid_numeric	books_borrowed_numeric	days_kept_numeric
fine_paid_numeric	1.000000	0.290422	0.383304
books_borrowed_numeric	0.290422	1.000000	0.096578
days_kept_numeric	0.383304	0.096578	1.000000

Borrowing mor↓ book↓ slightly affect↓ fin↓.

```
#29. Count how many students borrowed books in each month of 2025.
df["timestamp"] = pd.to_datetime(df["timestamp"], errors="coerce")
df_2025 = df[df["timestamp"].dt.year == 2025]
# Group by month number (1-12) and count
monthly_new_members = df_2025.groupby(df_2025["timestamp"].dt.month).size()
monthly_new_members
```

Analytical Statement:

```
timestamp
8      8
9     30
10    31
11    16
dtype: int64
```

Book↓ borrowing varie↓ across mor↓ th↓ in 2025. Som↓ month show high↓r student activity.

```
#30 .Create a pivot table:
Genre vs Borrow Status (count of students)
pd.pivot_table(
    df,
    index="book_genre_you_usually_read",
    columns="borrow_status",
    values="name_of_student_full_name",
    aggfunc="count"
)
```

book_genre_you_usually_read \ borrow_status	Overdue	Returned
Academic / Educational	7.0	5.0
Academic / Educational, Biography / Autobiography	NaN	1.0
Academic / Educational, Fiction, Self-help / Motivational, Biography / Autobiography	NaN	1.0
Academic / Educational, Historical	NaN	1.0

Analytical Statement: Acadamic book↓ show↓ mor↓ borrow and overdu↓ case↓. this helps analyz↓ reading and borrowing behaviour↓

Advanced Pandas Analysis Questions (31–50):

```
#31. Extract only the first name from "Name of Student (Full Name)".
df[["timestamp","name_of_student_full_name"]].head()
```

Analytical Statement:

	timestamp	name_of_student_full_name
0	Timestamp	Yash surve
1	Timestamp	sanika gole
2	Timestamp	saniya koli
3	Timestamp	Kalpana irle
4	Timestamp	Tushar bhosale

First names were extracted from full student names. this simplifies identification. makes data cleaner and easier to use.

```
#32. Create a new column Borrow_Duration_Category:
# Short (0-3 days)
# Medium (4-7 days)
# Long (8+ days)

df["Borrow_Duration_Category"] = pd.cut(
    pd.to_numeric(
        df["how_many_days_you_kept_the_books_?"].str.extract(r'(\d+)')[0],
        errors="coerce"
    ),
    bins=[-1,3,7,100],
    labels=["Short","Medium","Long"]
)
```

```
df[["how_many_days_you_kept_the_books_?","Borrow_Duration_Category"]].head()
```

	how_many_days_you_kept_the_books_?	Borrow_Duration_Category
0	4–7 days	Medium
1	8–14 days	Long
2	0–3 days	Short
3	0–3 days	Short
4	4–7 days	Medium

Analytical Statement: Borrow days were grouped into short, medium, and long. this helps classify borrowing behaviour.

```
#33. Clean and convert How many times you visit library per week? into numeric.
df["visit_freq"] = pd.to_numeric(
    df["how_many_times_you_visit_library_in_a_week?"].astype(str).str.extract(r'(\d+)')[0],
    errors="coerce"
)
```

Analytical Statement:

visit frequency was cleand and converted to number. text error were removed. numberic data accuray

```
df[["how_many_times_you_visit_library_in_a_week?","visit_freq"]].head()
```

	how_many_times_you_visit_library_in_a_week?	visit_freq
0	2–3 times	2.0
1	2–3 times	2.0
2	4–5 times	4.0
3	6+ times	6.0
4	4–5 times	4.0

```
#34. Clean "Average Study Hours per Week" column and convert into numeric values.
df["study_hours"] = pd.to_numeric(
    df["your_average_study_hours_in_library_per_week"]
        .astype(str)
        .str.extract(r'(\d+)')[0],
    errors="coerce"
)
```

```
df[["your_average_study_hours_in_library_per_week","study_hours"]].head()
```

	your_average_study_hours_in_library_per_week	study_hours
0	3–5 hours	3.0
1	3–5 hours	3.0
2	0–2 hours	0.0
3	9+ hours	9.0
4	3–5 hours	3.0

Analytical Statement: Study hours were cleaned and standardized. Converted into nnaimeric values. supports calculation

```
#35. Find whether students who visit library more often also borrow more books.
df["books_borrowed"] = pd.to_numeric(
    df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"]
        .astype(str)
        .str.extract(r'(\d+)')[0],
    errors="coerce"
)
```

Analytical Statement:

Frequent visitors tend to borrow mon books. shows a positive usage relations.

```
df[["number_of_books_you_borrowed_in_a_month_e.g._3,5,6","books_borrowed"]].head()
```

	number_of_books_you_borrowed_in_a_month_e.g._3,5,6	books_borrowed
0	1–2	1.0
1	1–3	1.0
2	1–4	1.0
3	1–5	1.0
4	1–6	1.0

```
#36. Find the percentage of students who paid a fine.
df["fine_paid_numeric"] = pd.to_numeric(
    df["fine_paid__₹"].astype(str),
    errors="coerce"
)

df[["fine_paid__₹","fine_paid_numeric"]].head()
```

Analytical Statement:

Students were marked as fined or not fined fine paid upto 100 to 150 mosty. helps track overdue behaviour.

	fine_paid__₹	fine_paid_numeric
0	0	0.0
1	100	100.0
2	0	0.0
3	150	150.0
4	100	100.0

```
#37. Add a new column Is_Fined (True/False).

# Convert number of books borrowed
df["books_borrowed"] = pd.to_numeric(
    df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"]
        .astype(str)
        .str.extract(r'(\d+)')[0],
   errors="coerce"
)
# Convert library visit frequency
df["visit_freq"] = pd.to_numeric(
    df["how_many_times_you_visit_library_in_a_week?"]
        .astype(str)
        .str.extract(r'(\d+)')[0],
    errors="coerce"
)
# Convert study hours
df["study_hours"] = pd.to_numeric(
    df["your_average_study_hours_in_library_per_week"]
        .astype(str)
        .str.extract(r'(\d+)')[0],
    errors="coerce"
)
# Convert fine paid
df["fine_paid_numeric"] = pd.to_numeric(
    df["fine_paid__₹"].astype(str),
    errors="coerce"
)
# Create Is_Fined column
df["Is_Fined"] = df["fine_paid_numeric"] > 0
# Calculate percentage of students who paid fine
fine_percentage = round(df["Is_Fined"].mean() * 100, 2)
df[["books_borrowed","visit_freq","study_hours","fine_paid_numeric","Is_Fined"]].head(), fine_percentage
```

```
(   books_borrowed  visit_freq  study_hours  fine_paid_numeric  Is_Fined
 0             1.0         2.0          3.0                0.0     False
 1             1.0         2.0          3.0              100.0      True
 2             1.0         4.0          0.0                0.0     False
 3             1.0         6.0          9.0              150.0      True
 4             1.0         4.0          3.0              100.0      True,
 62.39)
```

Analytical Statement: shows booked borrowed, visit frequency, Study hours and fined paid by collage student in library.

```
#38. Identify whether Premium students pay more fines than Standard members.
df.groupby("library_membership_type")["fine_paid_numeric"].mean()
```

```
library_membership_type
Premium     116.000000
Standard     89.183673
Name: fine_paid_numeric, dtype: float64
```

Analytical Statement: preminum membus fines wer compared with Standarder wers. helps valute membership impact

```
#39.  Group by Genre and calculate:
# average books borrowed
# average study hours

import numpy as np
def range_to_median(x):
    if pd.isna(x):
        return np.nan
    x = str(x).replace('–','-').replace(' ','')  # handle en-dash or spaces
    parts = x.split('-')
    nums = [float(p) for p in parts if p.isdigit()]
    return np.median(nums) if nums else np.nan

df["books_borrowed"] = df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"].apply(range_to_median)

df["study_hours"] = pd.to_numeric(
    df["your_average_study_hours_in_library_per_week"].astype(str).str.extract(r'(\d+)')[0],
    errors="coerce"
)
# Now group by genre
df.groupby("book_genre_you_usually_read")[["books_borrowed","study_hours"]].mean()
```

Analytical Statement: Avarage borred diffr by genr

book_genre_you_usually_read	books_borrowed	study_hours
Academic / Educational	2.857143	4.000000
Academic / Educational, Biography / Autobiography	1.500000	1.500000
Academic / Educational, Fiction, Self-help / Motivational, Biography / Autobiography	3.500000	6.000000

```
#40. Convert book ranges (1-2, 3-4) into median values and create a boxplot-ready numeric column.

import numpy as np

def range_to_median(x):
    if pd.isna(x):
        return np.nan
    x = str(x).replace('–','-').replace(' ','')  # replace en-dash / spaces
    parts = x.split('-')
    try:
        nums = [float(p) for p in parts if p.isdigit()]
        return np.median(nums) if nums else np.nan
    except:
        return np.nan

df["books_borrowed"] = df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"].apply(range_to_median)

df[["number_of_books_you_borrowed_in_a_month_e.g._3,5,6","books_borrowed"]].head()
```

	number_of_books_you_borrowed_in_a_month_e.g._3,5,6	books_borrowed
0	1–2	1.5
1	1–3	2.0
2	1–4	2.5
3	1–5	3.0
4	1–6	3.5

Analytical Statement: Book rangs were converted to median values makes data numeric and comparable.

```python
#41. Identify students who kept books longer than 10 days AND paid fine.
df["days_kept_numeric"] = pd.to_numeric(
    df["how_many_days_you_kept_the_books_?"].astype(str).str.extract(r'(\d+)')[0],
    errors="coerce"
)
# Filter students who kept books >10 days AND paid fine
long_kept_fined = df[
    (df["days_kept_numeric"] > 10) &
    (df["Is_Fined"])
]
long_kept_fined[["name_of_student_full_name","days_kept_numeric","fine_paid_numeric","Is_Fined"]]
```

Analytical Statement:

	name_of_student_full_name	days_kept_numeric	fine_paid_numeric	Is_Fined
25	Kishan yadav	15.0	100.0	True
31	Riya irle	15.0	250.0	True
57	Sharmila Tagore	15.0	250.0	True
68	Krunal karulkar	15.0	250.0	True
81	Omkar shinde	15.0	250.0	True
92	Trunal tandel	15.0	250.0	True

Student keeping booken longer often paid fines support monitoring

```python
#42. Create a new column Usage_Level based on:
# Study hours
# Visit frequency

df["Usage_Level"] = pd.cut(
    df["study_hours"].fillna(0) + df["visit_freq"].fillna(0),
    bins=[-1,6,12,100],
    labels=["Low","Medium","High"]
)
df[["study_hours","visit_freq","Usage_Level"]].head()
```

Analytical Statement:

Student classfied as high, medium or low users.

Based on visits and study time.

	study_hours	visit_freq	Usage_Level
0	3.0	2.0	Low
1	3.0	2.0	Low
2	0.0	4.0	Low
3	9.0	6.0	High
4	3.0	4.0	Medium

```python
#43. Find if Historical readers study more hours than Fiction readers.
df[df["book_genre_you_usually_read"].isin(["Historical","Fiction"])] \
    .groupby("book_genre_you_usually_read")["study_hours"].mean()
```

```
book_genre_you_usually_read
Fiction       4.50
Historical    4.65
Name: study_hours, dtype: float64
```

Analytical Statement: Historical readers study slightly longer. Reading Preference affects habits.

```
#44. Group by joining month (Jan-Dec) and count number of new members
df["timestamp"] = pd.to_datetime(df["timestamp"], errors="coerce")
df["Join_Month"] = df["timestamp"].dt.month_name()
# Count new members per month
df.groupby("Join_Month").size()
```

```
Join_Month
August         8
November      16
October       31
September     30
dtype: int64
```

Analytical Statement: new memberships vary by moth. shows enrollment trends. helps planning

```
#45. Rank all students by:
# Fine Paid
# Books Borrowed

df["fine_paid_numeric"] = pd.to_numeric(df["fine_paid__?"].astype(str), errors="coerce")
df["books_borrowed"] = pd.to_numeric(
    df["number_of_books_you_borrowed_in_a_month_e.g._3,5,6"].astype(str).str.extract(r'(\d+)')[0],
    errors="coerce"
)
df["Fine_Rank"] = df["fine_paid_numeric"].rank(ascending=False)
df["Books_Rank"] = df["books_borrowed"].rank(ascending=False)

df[["fine_paid_numeric","Fine_Rank","books_borrowed","Books_Rank"]].head()
```

Analytical Statement:

Extreme fin values were identified. prevents misleading analysis

	fine_paid_numeric	Fine_Rank	books_borrowed	Books_Rank
0	0.0	84.0	1.0	77.5
1	100.0	54.5	1.0	77.5
2	0.0	84.0	1.0	77.5
3	150.0	29.5	1.0	77.5
4	100.0	54.5	1.0	77.5

```
#46. Create a column Late_Return_Risk based on:
# Study hours
#Visiting frequency
# Books borrowed
days_kept = pd.to_numeric(
    df["how_many_days_you_kept_the_books_?"].astype(str).str.extract(r'(\d+)')[0],
    errors="coerce"
)
df["Late_Return_Risk"] = (
    (df["borrow_status"] == "Overdue") &
    (df["books_borrowed"] > 3) &
    (days_kept > 7)
)
df[["borrow_status","books_borrowed","how_many_days_you_kept_the_books_?","Late_Return_Risk"]].head()
```

	borrow_status	books_borrowed	how_many_days_you_kept_the_books_?	Late_Return_Risk
0	Returned	1.5	4–7 days	False
1	Overdue	2.0	8–14 days	False
2	NaN	2.5	0–3 days	False
3	Overdue	3.0	0–3 days	False
4	Overdue	3.5	4–7 days	False

Analytical Statement:
Student ranked by fines and borrowing. High activity users identified

```
#48. Compare FY vs SY vs TY in:
# Study hours
# Visiting frequency
# Books borrowed
df.groupby("year_of_study")[["study_hours","visit_freq","books_borrowed"]].mean()
```

Analytical Statement:

year_of_study	study_hours	visit_freq	books_borrowed
FY	3.600000	2.160000	2.50
Other	9.000000	6.000000	7.00
SY	4.170732	2.487805	2.65
TY	4.968750	2.687500	3.60

Senior students show higher usage.
Visits and borrowing rises with year

```
#49. Calculate cumulative total fine collected month-wise.
df["timestamp"] = pd.to_datetime(df["timestamp"], errors="coerce")
df["Join_Month"] = df["timestamp"].dt.month_name()
df.groupby("Join_Month")["fine_paid_numeric"].sum().cumsum()
```

```
Join_Month
August         800.0
November      1250.0
October       4600.0
September     7900.0
Name: fine_paid_numeric, dtype: float64
```

Analytical Statement: monthly fine totals were calculated. shows upward trend. Shows revenue trend

```
#50. Predict (rule-based, not ML): Which students are likely to become high-frequency borrowers based on past behavior.
df["High_Frequency_Predict"] = (
    (df["library_visits_numeric"] >= 4) &
    (df["books_borrowed_numeric"] >= 4)
)

df["High_Frequency_Predict"].value_counts()
```

```
High_Frequency_Predict
False    92
True     17
Name: count, dtype: int64
```

Analytical Statement: Past behaviour predicts future borrowing. Active users likely remain consistent. helps resource planning.

SWOT Analysis – College Library Data Usage

Strengths

- Uses real college library data for analysis
- Covers basic, intermediate, and advanced Pandas operations
- Helps understand student behavior and library usage patterns
- Proper data cleaning and feature extraction improves accuracy
- Clear classification of students based on usage level

Weaknesses

- Dataset is limited to one college only
- Some columns required cleaning due to text-based values
- No visual dashboards included
- Analysis depends on accuracy of student responses

Opportunities

- Results can help improve library services and planning
- Can be extended with data visualization dashboards
- Useful for predicting high-frequency library users
- Can support decision-making for membership benefits
- Future scope to apply Machine Learning models

Threats

- Incomplete or incorrect data can affect results
- Changing student behavior may reduce accuracy over time
- Manual data entry can introduce errors
- Privacy and data security concerns

Conclusion

This project, *"College Library Data Usage Analysis"*, provides a clear understanding of how students interact with the college library and how different factors influence library usage patterns. By applying Python and Pandas, the project successfully transforms raw library data into meaningful insights.

Library Usage Patterns

- The analysis shows noticeable differences in library usage across departments and years of study.
- Senior students (SY/TY) generally demonstrate higher borrowing frequency and study hours compared to first-year students.
- Students with Premium membership tend to use library facilities more actively than Standard members.

Borrowing Behavior and Study Habits

- A positive relationship is observed between library visit frequency, study hours, and number of books borrowed.
- Students who visit the library more often are likely to borrow more books and spend longer study hours.
- Certain book genres, such as academic and self-help categories, are more popular among regular library users.

Fine Patterns and Overdue Behavior

- Fine analysis highlights that overdue returns are a major contributor to fine collection.
- Students who keep books for longer durations show a higher risk of paying fines.
- Identifying such patterns helps in understanding late return behavior and improving library policies.

Data Analysis and Technical Learning

- The project demonstrates effective use of data cleaning, transformation, grouping, and aggregation techniques.
- Advanced operations such as feature engineering, classification, correlation analysis, and rule-based prediction enhance analytical depth.
- The step-by-step analysis strengthens practical knowledge of real-world data handling using Pandas.

Overall Impact

- This analysis proves that structured data analysis can support better academic and administrative decision-making.
- The project not only improves understanding of student learning behavior but also highlights areas for improving library management.
- institutional improvement.