Movie Ratings Dataset Exploration with Pivot Tables

Project Description:

- This project analyzes a movie ratings dataset using Python (Pandas) and builds an interactive web application with Streamlit. The system provides insights into user preferences, most popular movies, genre performance, and ratings distribution. It makes use of pivot tables, grouping, filtering, and aggregation. Additional features include exporting results into CSV files and generating visual insights with charts.

The interactive webpage allows users to:

- Filter by genre, user, or rating range.
- Explore pivot tables (average ratings, counts, etc.).
- Visualize results with bar charts and pie charts.
- Export cleaned datasets and pivot tables for further use.

BY:- Name: Unnati M P

Institution: GSSS Institute of Engineering and Technology for Women

Batch: 2023 – 2027

Branch: Artificial Intelligence and Data Science (AI&DS)

Index

Detailed Explanation

Overview

The project demonstrates the use of pivot tables and visualizations in analyzing movie ratings data. It integrates data from Movies, Ratings, and Users datasets to produce meaningful insights. The **Movie Rating Analysis** project is designed to explore and analyze movie rating datasets by leveraging **Python (Pandas, Matplotlib, Seaborn)** and an interactive **Streamlit web application**. The system integrates three key datasets — **movies**, **users**, and **ratings** — to provide comprehensive insights into audience preferences, genre popularity, and rating behavior.

The project begins with **data preprocessing**, where missing values are handled, duplicates are removed, and datasets are merged into a single cleaned dataset. Next, derived attributes such as **Rating Categories (High/Medium/Low)** are introduced to enhance analysis.

The application implements multiple **pivot tables and visualizations** that highlight important use-cases, including:

- Average rating per movie, genre, and user.
- Count of ratings per movie to identify popular titles.
- Distribution of ratings across categories (High/Medium/Low).
- Interactive filtering by genre, user, and rating range.

📖 Use-Case Explanations for *Movie Rating Analysis Project*

This section provides a detailed explanation of the major **use-cases** implemented in the project. Each use-case highlights the **goal**,and **outcome**, supported by pivot tables, visualizations, and export features.

🎥 Use-Case 1: Average Rating per Movie

- **Goal**: To compute the mean rating of each movie and understand which movies are most loved by viewers.
- **Methodology**:
  - A **pivot table** is created with `Title` as the index and the average of `Rating` as values.
  - A bar chart is used to visualize the ratings per movie.
- **Outcome**:
  - Enables identification of movies with consistently high audience approval.
  - Helps filter movies by average rating ≥ 4 to highlight critically appreciated titles.

🎭 Use-Case 2: Average Rating per Genre

- **Goal**: To analyze how different genres perform in terms of viewer ratings.
- **Methodology**:
  - A **pivot table** groups movies by `Genre` and calculates the average rating.
  - Visualization uses a **horizontal bar chart** with genres on the Y-axis.
- **Outcome**:
  - Helps discover the most popular genres.
  - Useful for recommendations, such as suggesting genres with higher engagement and satisfaction.

👤 Use-Case 3: Average Rating per User

- **Goal**: To study user behavior in rating movies and check for bias or consistency.
- **Methodology**:
  - A pivot table groups by `UserID` to calculate each user's average given rating.
  - A **line chart** visualizes variations among users.
- **Outcome**:
  - Helps identify generous users (who often rate higher) and critical users (who rate lower).
  - Assists in detecting rating anomalies for further recommendation refinement.

Use-Case 4: Count of Ratings per Movie (Most Rated)

- **Goal**: To identify movies that received the highest number of ratings, representing popularity.
- **Methodology**:
  - A pivot table counts the number of ratings per `Title`.
  - A **bar plot** (Top 10) visualizes the most-rated movies.
- **Outcome**:
  - Indicates popularity, even if not always high-rated.
  - Useful for identifying trending or widely viewed movies.

Use-Case 5: Ratings Distribution (High/Medium/Low)

- **Goal**: To categorize ratings into `High` (≥4), `Medium` (≥3 and <4), and `Low` (<3) and understand their distribution.
- **Methodology**:
  - A new column `RatingCategory` classifies each rating.
  - A **pie chart** displays the percentage distribution of these categories.
- **Outcome**:
  - Provides insights into overall satisfaction trends.
  - Shows whether the dataset is skewed toward positive or negative ratings.

Use-Case 6: Export of Data and Results

- **Goal**: To allow users to save processed results for further use.
- **Methodology**:
  - Export buttons are implemented to save pivot tables and cleaned datasets (`CSV` format).
  - Available exports include:
    - `movie_avg_ratings.csv`
    - `genre_avg_ratings.csv`
    - `user_avg_ratings.csv`
    - `movie_rating_counts.csv`
    - `ratings_distribution.csv`
    - `cleaned_movie_ratings.csv`
- **Outcome**:
  - Enhances usability for researchers and analysts.
  - Provides offline access for deeper study or reporting.

⚙ Algorithms / Approach

The **Movie Rating Analysis** applies a structured **data analytics pipeline** with systematic steps to process, analyze, and visualize the dataset. The approach can be summarized as follows:

---

1. Data Collection & Loading

- Three datasets were provided:
  - **movies.csv** → Contains movie titles and genres.
  - **ratings.csv** → Contains user ratings for movies.
  - **users.csv** → Contains user information (e.g., UserID, demographics if available).
- These datasets were imported into the system using **Pandas** for efficient handling of structured data.

---

2. Data Preprocessing

- **Merging Datasets**:
  - First, `ratings.csv` was merged with `movies.csv` using `MovieID`.
  - Then, the result was merged with `users.csv` using `UserID`.
  - This produced a **cleaned and unified dataset** containing movies, ratings, and user details.
- **Handling Missing Values**:
  - Checked for null values in all datasets.
  - Retained `NaN` values for ratings if found (optional fill with 0).
- **Removing Duplicates**: Ensured each `(UserID, MovieID)` pair is unique.

---

3. Feature Engineering

- Introduced a derived column **`RatingCategory`**:
  - **High** → Ratings ≥ 4
  - **Medium** → Ratings = 3
  - **Low** → Ratings ≤ 2
- This categorization allowed easy visualization of audience sentiment.

---

4. Analytical Use-Cases (Pivot Tables & Grouping)

Implemented **pivot tables** and **group-by operations** to support the following insights:

1. **Average Rating per Movie** → Identifies overall reception of each film.
2. **Average Rating per Genre** → Shows which genres are more liked by users.
3. **Average Rating per User** → Tracks rating behavior of individuals.
4. **Count of Ratings per Movie** → Identifies most-rated (popular) movies.

---

5. Visualization

- Used **Seaborn** and **Matplotlib** for visually appealing charts.
- Plots included:
  - **Bar Charts** → For Top 5 movies, genre averages, and most-rated movies.
  - **Pie Chart** → For distribution of High/Medium/Low ratings.
  - **Line Chart** → For average rating per user.
- Adopted **color palettes (crest, viridis, mako)** for aesthetic consistency.

---

6. Interactivity (Streamlit Integration)

- **Sidebar Filters** → Genre, User, Rating Range.
- **Expandable Previews** → Raw datasets preview.
- **Export Options** → Ability to save pivot tables and final cleaned dataset as CSV.
- **Logo Integration** → A custom Movie Rating Analysis logo added for branding.

---

7. Output & Export

- Exported CSV files for:
  - `movie_avg_ratings.csv`
  - `genre_avg_ratings.csv`
  - `user_avg_ratings.csv`
  - `movie_rating_counts.csv`
  - `cleaned_movie_ratings.csv`
- Ensured project outputs can be reused for external analysis and reporting.

UML Diagrams

1. Use-Case Diagram

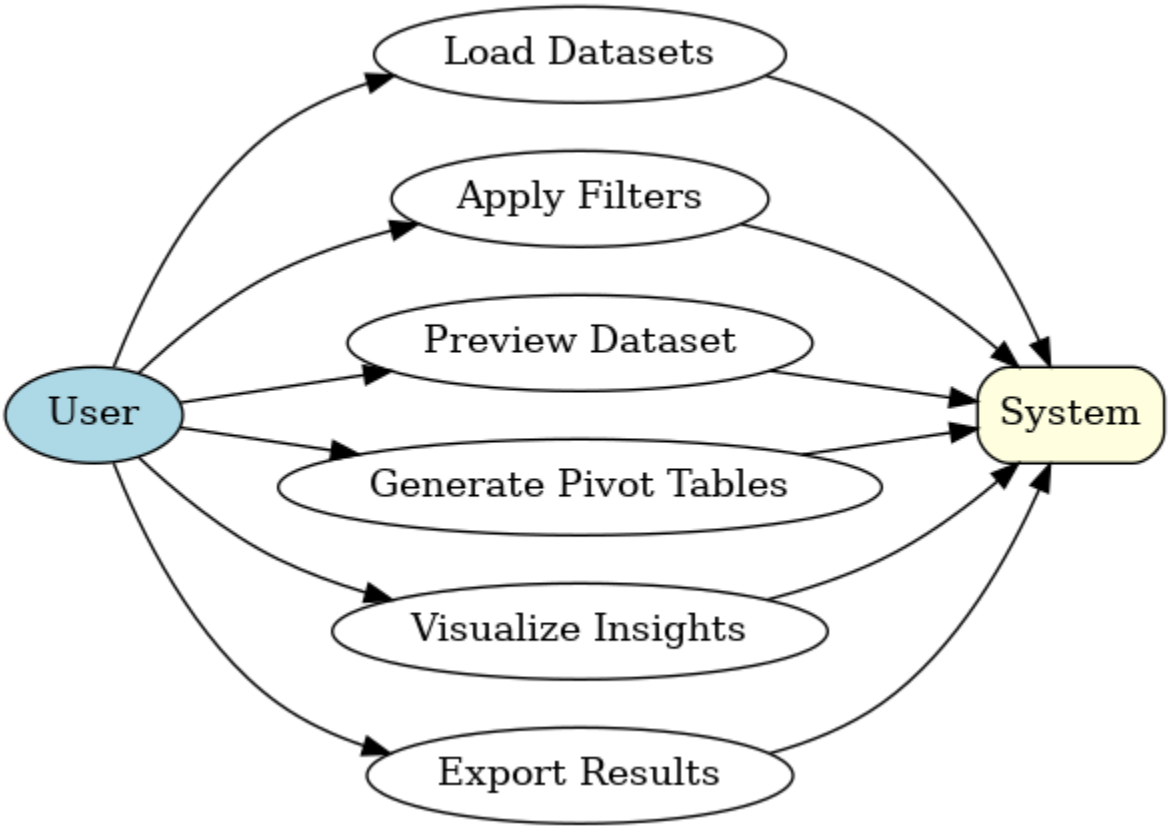
This diagram shows the **interactions between the user and the system**.

Actors:

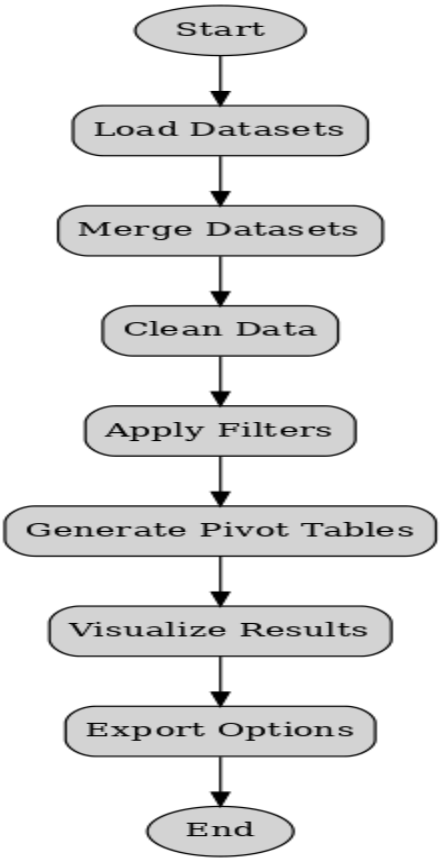
- **End User (Student/Analyst)** → explores ratings, applies filters, exports results.
- **System (Streamlit Web App)** → loads, merges, analyzes, visualizes, and exports datasets.

Main Use-Cases:

- Upload/Load Datasets (movies, ratings, users)
- Apply Filters (genre, user, rating range)
- View Dataset Preview
- Generate Pivot Tables (avg rating per movie/genre/user, rating counts)
- Visualize Insights (bar charts, pie charts, line charts)
- Export Results (CSV files, cleaned dataset)



2. Activity Diagram



Represents the **workflow of your system** (step-by-step data flow).

Flow:

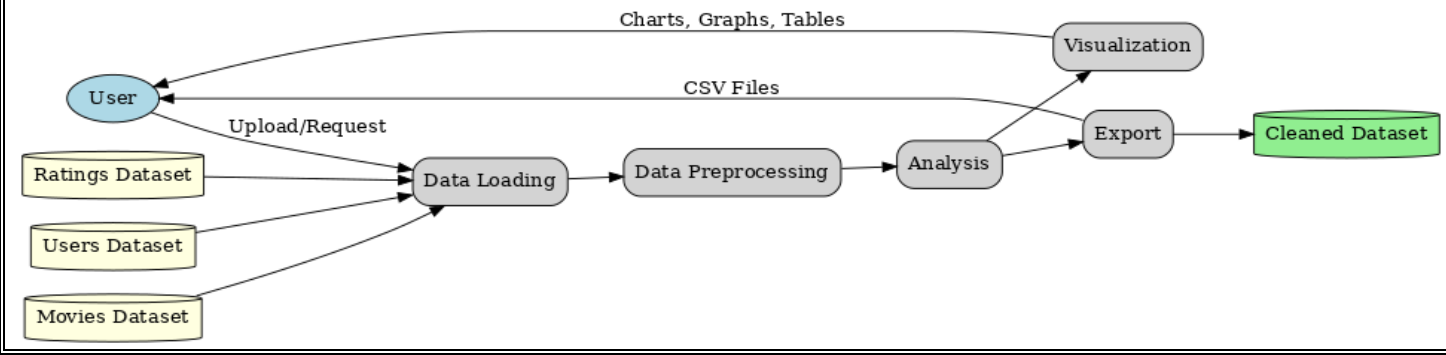
1. Start
2. Load Datasets (movies, ratings, users)
3. Merge Datasets (ratings + movies + users)
4. Data Cleaning (check missing values, duplicates, derive `RatingCategory`)
5. Apply Filters (genre, user, rating range)
6. Generate Pivot Tables
7. Visualize Results (charts & graphs)
8. Export Options (CSV, cleaned dataset)
9. End

3. Data Flow Diagram

Shows how **data moves** between components.

Entities:

- **User** → Provides input (filters, dataset selection).
- **System Modules:**
  - Data Loading Module (reads CSVs)
  - Data Preprocessing Module (clean, merge, transform)
  - Analysis Module (pivot tables, grouping)
  - Visualization Module (charts & graphs)
  - Export Module (CSV output).
- **Datastores:**
  - Movies Dataset
  - Ratings Dataset
  - Users Dataset
  - Cleaned & Processed Dataset



Front-End (Interface) Design

Overview

The front-end of this project is implemented using **Streamlit**, which provides an interactive web-based interface. It allows users to:

- Upload and preview datasets (`movies.csv`, `ratings.csv`, `users.csv`).
- Filter movies by genre, user, and rating range.
- View pivot tables and visual insights (average ratings, most-rated movies, etc.).
- Export processed results into CSV files.
- Visualize insights using bar charts, pie charts, and line charts.

Key UI Components

1. **Title**
   - Project title: *" 🎬 Movie Ratings Dataset Exploration with Pivot Tables"*
2. **Dataset Preview Section**
   - Expandable panels (`st.expander`) to view:
     - Movies dataset
     - Ratings dataset
     - Users dataset
3. **Sidebar Filters**
   - Genre selection (`selectbox`)
   - User selection (`selectbox`)
   - Rating range slider (`slider`)
4. **Interactive Insights Section**
   - Movies with average rating ≥ 4.0
   - Top 5 movies by ratings (table + bar chart)
   - Active users (who rated more than 5 movies)
   - Highest & lowest rated movies
5. **Pivot Tables Section**
   - Average rating per movie
   - Average rating per genre
   - Average rating per user
   - Count of ratings per movie
6. **Visual Insights Section**
   - Bar charts for top-rated movies
   - Pie chart for rating distribution (High/Medium/Low)
   - Bar chart for genre-wise averages
7. **Export Options**
   - Export individual pivot tables
   - Export filtered dataset
   - Export full cleaned dataset

SETUP INSTUCTIONS

📁 Project Structure

movie_ratings_project/

│

├── analysis.py # Data analysis with Pandas

├── app.py # Streamlit interactive app

├── movies.csv # Sample dataset (movies)

├── ratings.csv # Sample dataset (ratings)

├── movie_avg_ratings.csv # Exported pivot table (movies)

├── genre_avg_ratings.csv # Exported pivot table (genres)

├── user_avg_ratings.csv # Exported pivot table (users)

├── cleaned_movie_ratings.csv # Final cleaned + merged dataset

├── requirements.txt # Python dependencies

└── README.md # Project documentation

SETTING UP START:

1️⃣ Clone or Download the Project

Unzip the project folder or clone it from GitHub

2️⃣ Setup Virtual Environment

Open terminal inside the project folder and run:

python -m venv venv

venv\Scripts\activate # Windows

source venv/bin/activate # Mac/Linux

3️⃣ Install Dependencies

pip install -r requirements.txt

4️⃣ Run Data Analysis (Optional)

python analysis.py

5️⃣ Run the Web App

streamlit run app.py

🛠️ Technologies Used

- Python 3.12
- Pandas
- Streamlit
- Matplotlib & Seaborn

Code

Click here to view full source code: app_code.txt
-ctrl+click to follow the code

Explanation of the Code

The project is implemented in **Python** using the **Streamlit framework** for building an interactive web application. It integrates **Pandas** for data handling, **Seaborn/Matplotlib** for visualization.

Below is the structured explanation with code snippets.

---

1. Importing Libraries

```
import pandas as pd
import streamlit as st
import matplotlib.pyplot as plt
import seaborn as sns
```

- **pandas** → Data manipulation and analysis.
- **streamlit** → Web application framework for interactive dashboards.
- **matplotlib.pyplot** & **seaborn** → For data visualization and prettier charts.

---

2. Loading Data

```
movies = pd.read_csv("movies.csv")
ratings = pd.read_csv("ratings.csv")
users = pd.read_csv("users.csv")
```

- `movies.csv`: Movie details (titles, genres).
- `ratings.csv`: Ratings provided by users.
- `users.csv`: User demographic details.

Data is loaded into DataFrames for further processing.

---

3. Merging Datasets

```
ratings_movies = pd.merge(ratings, movies, on="MovieID", how="inner")
merged = pd.merge(ratings_movies, users, on="UserID", how="inner")
```

- **Step 1**: Merge `ratings` with `movies` on `MovieID`.
- **Step 2**: Merge the result with `users` on `UserID`.
- Final `merged` DataFrame contains user, movie, rating, and genre details in one place.

---

4. Creating Derived Columns

```
merged["RatingCategory"] = merged["Rating"].apply(
```

```
    lambda x: "High" if x >= 4 else ("Medium" if x >= 3 else "Low")
)
```

- Ratings are categorized as:
  - **High** (≥ 4)
  - **Medium** ($3 \leq$ rating < 4)
  - **Low** (< 3)
- This helps in analyzing overall distribution.

---

5. Pivot Tables (Pre-Computed Insights)

```
movie_rating_counts = (
    merged.pivot_table(index="Title", values="Rating", aggfunc="count")
    .reset_index()
    .rename(columns={"Rating": "RatingCount"})
)

movie_avg_ratings = (
    merged.pivot_table(index="Title", values="Rating", aggfunc="mean")
    .reset_index()
    .rename(columns={"Rating": "AvgRating"})
)

movie_rating_counts["IsPopular"] = movie_rating_counts["RatingCount"].apply(
    lambda x: "Yes" if x > 10 else "No"
)
```

- **Count of Ratings per Movie** → How many times each movie was rated.
- **Average Rating per Movie** → Mean rating for each movie.
- **Popularity Flag** → Movies with more than 10 ratings are labeled as "Yes" (popular).

---

6. Streamlit UI Setup

```
st.title("🎬 Movie Ratings Dataset Exploration with Pivot Tables")

st.header("Dataset Preview")
with st.expander("Show Movies Dataset"):
    st.dataframe(movies)
with st.expander("Show Ratings Dataset"):
    st.dataframe(ratings)
with st.expander("Show Users Dataset"):
    st.dataframe(users)
```

- The app displays a **title** and expandable dataset previews.
- Users can quickly explore raw data without exporting it.

---

7. Filters (Sidebar Controls)

```
st.sidebar.header("🔎 Filters")
genres = movies["Genre"].unique().tolist()
selected_genre = st.sidebar.selectbox("Select Genre", ["All"] + genres)

users_list = merged["UserID"].unique().tolist()
selected_user = st.sidebar.selectbox("Select User", ["All"] + users_list)

min_rating, max_rating = st.sidebar.slider(
    "Select Rating Range", 1.0, 5.0, (1.0, 5.0)
)
```

- **Genre filter** → Select specific genres or all.
- **User filter** → Focus on ratings by a specific user.
- **Rating range filter** → Restrict ratings between chosen values.

👉 These filters update the dataset dynamically.

---

8. Filtered Dataset

```
filtered_data = merged.copy()

if selected_genre != "All":
    filtered_data = filtered_data[filtered_data["Genre"] == selected_genre]

if selected_user != "All":
    filtered_data = filtered_data[filtered_data["UserID"] == selected_user]

filtered_data = filtered_data[
    (filtered_data["Rating"] >= min_rating) & (filtered_data["Rating"] <=
max_rating)
]

st.header("🎯 Filtered Dataset")
st.dataframe(filtered_data.head(20))
```

- Creates a copy of merged dataset.
- Applies user-selected filters.
- Displays first 20 rows in a table format.

---

9. Interactive Insights

a) Movies with Avg Rating ≥ 4.0

```
highly_rated = filtered_data.groupby("Title")["Rating"].mean().reset_index()
highly_rated = highly_rated[highly_rated["Rating"] >= 4.0]
st.dataframe(highly_rated)
```

Shows movies with strong user approval.

---

b) Top 5 Movies by Ratings (Pivot + Chart)

```
top5 = (
    merged.groupby("Title")["Rating"]
    .count()
    .sort_values(ascending=False)
    .head(5)
    .reset_index()
    .rename(columns={"Rating": "RatingCount"})
)
```

- Pivot table for **Top 5 most rated movies**.
- Displayed in **table** and **bar chart** with annotation labels.

---

c) Highest & Lowest Rated Movies

```
avg_ratings = filtered_data.groupby("Title")["Rating"].mean().reset_index()
highest = avg_ratings.sort_values(by="Rating", ascending=False).head(1)
lowest = avg_ratings.sort_values(by="Rating", ascending=True).head(1)
```

Displays extremes in ratings.

---

10. Pivot Tables & Graphs

- **Average Rating per Movie**

```
pivot_movie_avg = merged.pivot_table(index="Title", values="Rating",
aggfunc="mean").reset_index()
st.bar_chart(pivot_movie_avg.set_index("Title"))
```

- **Average Rating per Genre**

```
pivot_genre_avg = merged.pivot_table(index="Genre", values="Rating",
aggfunc="mean").reset_index()
sns.barplot(x="Rating", y="Genre", data=pivot_genre_avg, palette="viridis")
```

- **Average Rating per User**

```
pivot_user_avg = merged.pivot_table(index="UserID", values="Rating",
aggfunc="mean").reset_index()
st.line_chart(pivot_user_avg.set_index("UserID"))
```

- **Count of Ratings per Movie**

```
pivot_movie_count = (
    merged.pivot_table(index="Title", values="Rating", aggfunc="count")
    .reset_index()
    .rename(columns={"Rating": "RatingCount"})
)
sns.barplot(x="RatingCount", y="Title",
data=pivot_movie_count.sort_values(by="RatingCount",
ascending=False).head(10))
```

---

11. Visual Insights (Pie Chart)

```
rating_dist = merged["RatingCategory"].value_counts()
colors = sns.color_palette("crest", len(rating_dist))

ax1.pie(
    rating_dist,
    labels=rating_dist.index,
    autopct="%1.1f%%",
    startangle=90,
    colors=colors,
    wedgeprops={"edgecolor": "white"}
)
```

- Shows distribution of High, Medium, Low ratings in a clean pie chart.

---

12. Export Options

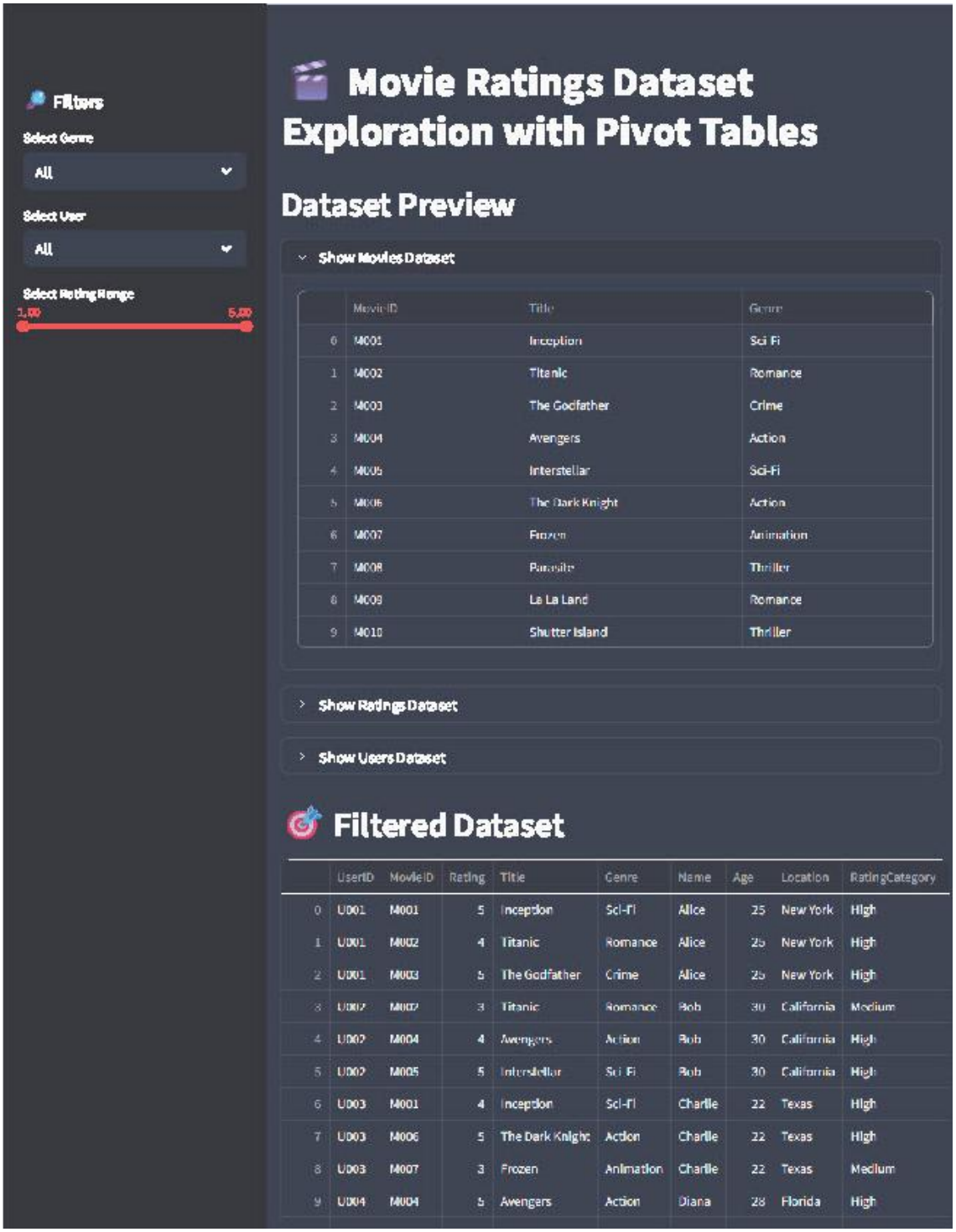
The app provides CSV exports for every major analysis:

```
top5.to_csv("top5_movies_by_ratings.csv", index=False)
pivot_movie_avg.to_csv("movie_avg_ratings.csv", index=False)
pivot_genre_avg.to_csv("genre_avg_ratings.csv", index=False)
pivot_user_avg.to_csv("user_avg_ratings.csv", index=False)
pivot_movie_count.to_csv("movie_rating_counts.csv", index=False)
merged.to_csv("cleaned_movie_ratings.csv", index=False)
```

This ensures **reusability** of results in Excel

Screenshots of the Output

Screenshots of the dashboard, graphs, and exported files will be attached here.



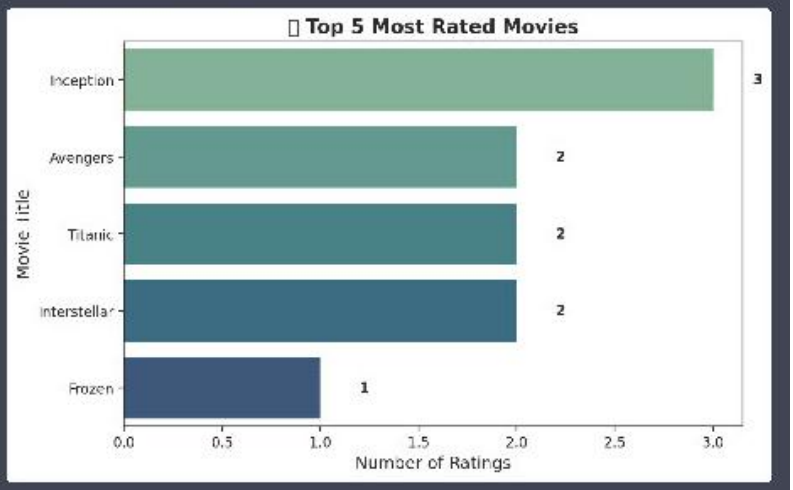
Movies with Average Rating ≥ 4.0

	Title	Rating
0	Avengers	4.5
3	Interstellar	4.5
4	La La Land	4
5	Parasite	5
7	The Dark Knight	5
8	The Godfather	5

Top 5 Movies by Number of Ratings

	Title	RatingCount
0	Inception	3
1	Avengers	2
2	Titanic	2
3	Interstellar	2
4	Frozen	1

Export Top 5 Movies by Ratings



4. Highest & Lowest Rated Movies

Highest Rated

	Title	Rating
7	The Dark Knight	5

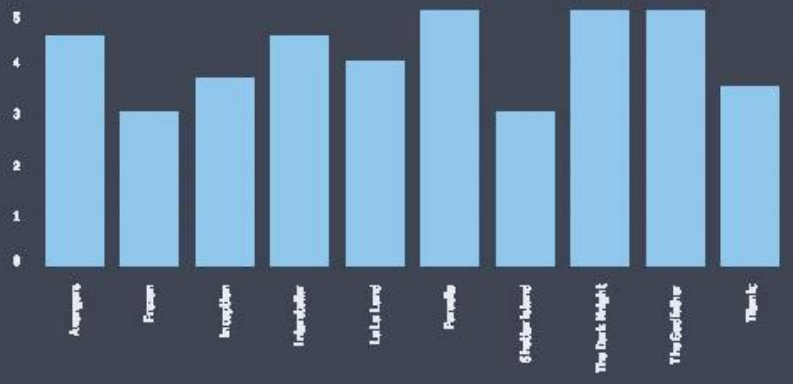
Lowest Rated

	Title	Rating
1	Frozen	3

Average Rating per Movie

	Title	Rating
0	Avengers	4.5
1	Frozen	3
2	Inception	3.6667
3	Interstellar	4.5
4	La La Land	4
5	Parasite	5
6	Shutter Island	3
7	The Dark Knight	5
8	The Godfather	5
9	Titanic	3.5

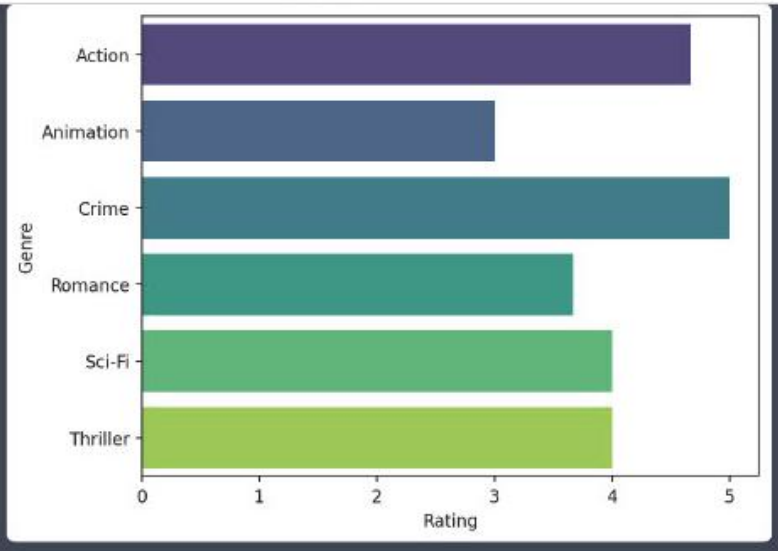
Export Avg Rating per Movie



Average Rating per Genre

	Genre	Rating
0	Action	4.6667
1	Animation	3
2	Crime	5
3	Romance	3.6667
4	Sci-Fi	4
5	Thriller	4

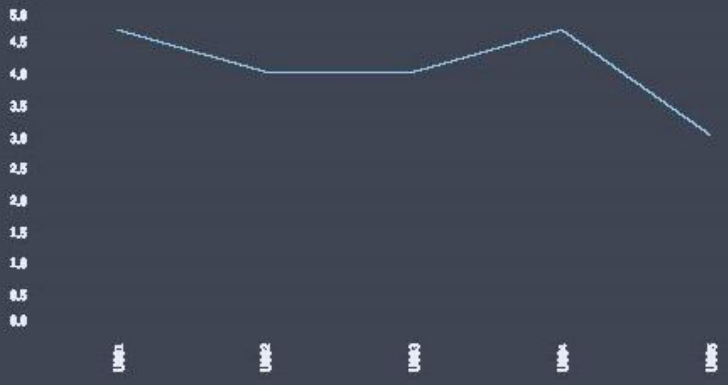
Export Avg Rating per Genre



Average Rating per User

	UserID	Rating
0	U001	4.6667
1	U002	4
2	U003	4
3	U004	4.6667
4	U005	3

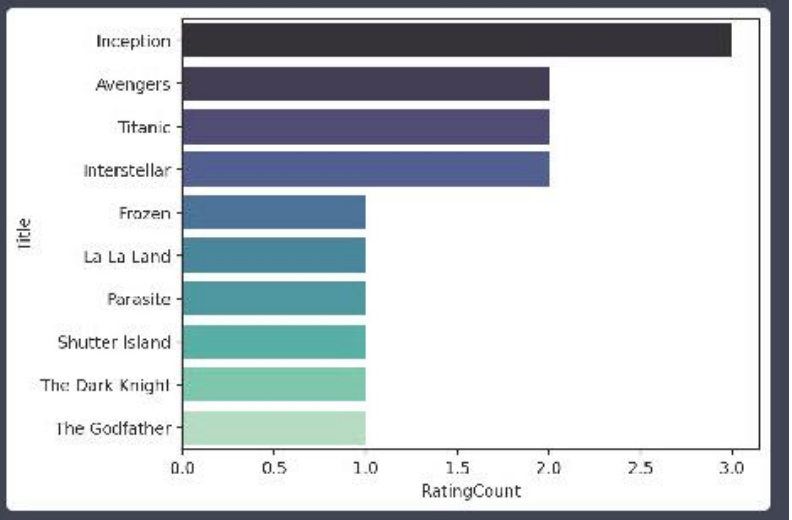
Export Avg Rating per User



Count of Ratings per Movie (Most Rated)

	Title	RatingCount
2	Inception	3
0	Avengers	2
9	Titanic	2
3	Interstellar	2
1	Frozen	1
4	La La Land	1
5	Parasite	1
6	Shutter Island	1
7	The Dark Knight	1
8	The Godfather	1

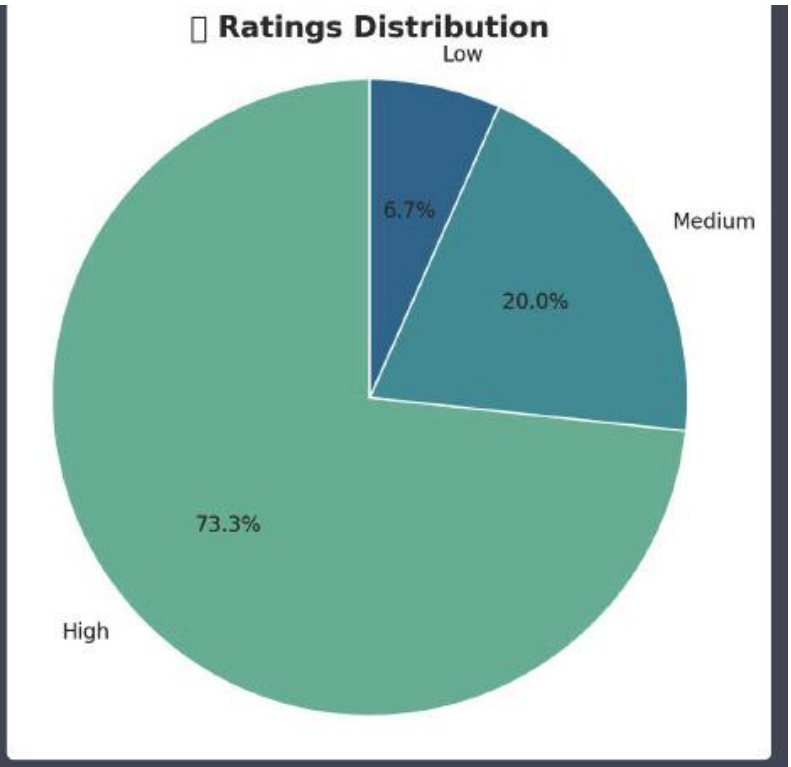
Export Count of Ratings per Movie



Ratings Distribution (High / Medium / Low)

	Count	count
0	High	11
1	Medium	3
2	Low	1

Export Ratings Distribution



Export Cleaned & Merged Dataset

	UserID	MovieID	Rating	Title	Genre	Name	Age	Location	RatingCategory
0	U001	M001	5	Inception	Sci-Fi	Alice	25	New York	High
1	U001	M002	4	Titanic	Romance	Alice	25	New York	High
2	U001	M003	5	The Godfather	Crime	Alice	25	New York	High
3	U002	M002	3	Titanic	Romance	Bob	30	California	Medium
4	U002	M004	4	Avengers	Action	Bob	30	California	High
5	U002	M005	5	Interstellar	Sci-Fi	Bob	30	California	High
6	U003	M001	4	Inception	Sci-Fi	Charlie	22	Texas	High
7	U003	M006	5	The Dark Knight	Action	Charlie	22	Texas	High
8	U003	M007	3	Frozen	Animation	Charlie	22	Texas	Medium
9	U004	M004	5	Avengers	Action	Diana	28	Florida	High

Export Cleaned Movie Ratings Dataset

Closure (Conclusion & Bibliography)

Conclusion:
The project successfully demonstrates the use of Pandas and Streamlit to perform dataset exploration interactively. By using pivot tables, grouping, and aggregation, users can identify key insights like popular movies, top-rated genres, and user activity. The Streamlit interface makes it user-friendly and professional.

Bibliography:

- Python Pandas Documentation
- Streamlit Documentation
- Matplotlib & Seaborn Libraries
- Dataset inspired by MovieLens

Dataset: Sample dataset prepared for analysis.
Tools: VS Code, Streamlit, Draw.io (for UML diagrams).