Session-Based Cross Domain Recommendation System

Intro to Data Science, CAP5771 - Project

Santhi Daggubati

UFID: 65589133

Milestone 1: Data Collection & Preprocessing

Data Collection:

Three datasets were used of different domains: Movies, Music & Books from ***Hugging Face*** Dataset collection.

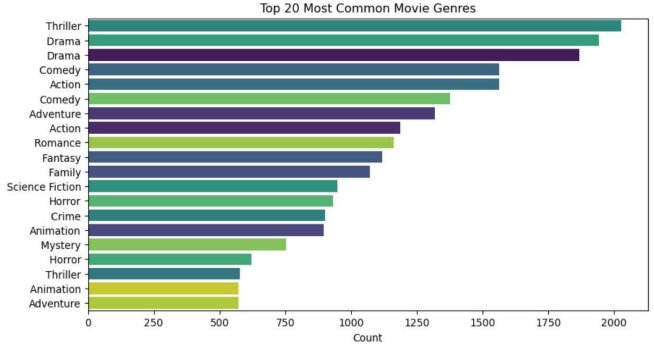
- Movies: IMDb Dataset
- Music: Spotify Tracklist Dataset
- Books: GoodReads Dataset

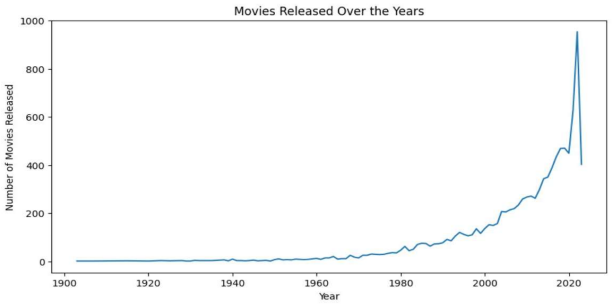
Preprocessing:

- **Unifying Datasets:** Assigned **unique Item_IDs** to each domain (Movies, Music, Books) by prefixing IDs with respective content type (`Movie_`, `Music_`, `Book_`).
- **Renaming for Consistency:** `orig_title` → `Title`, `genre` → `Genre`
- **Handled Missing Data**(filling and dropping), **Standardizing Columns**(consistent columns across all datasets), **Adding Item_type column**(to specify content type: "Movie", "Music", or "Book")

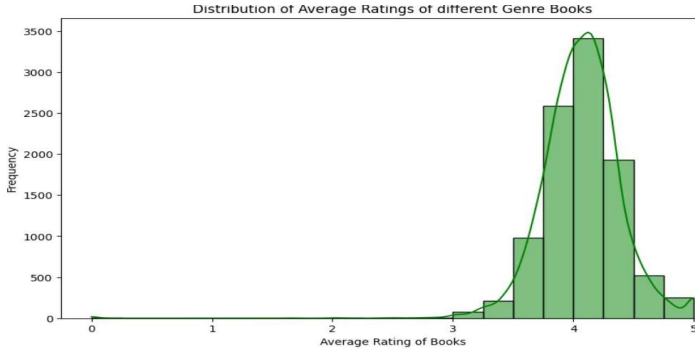
Milestone 1: Visualizations

Movies Dataset Trends:

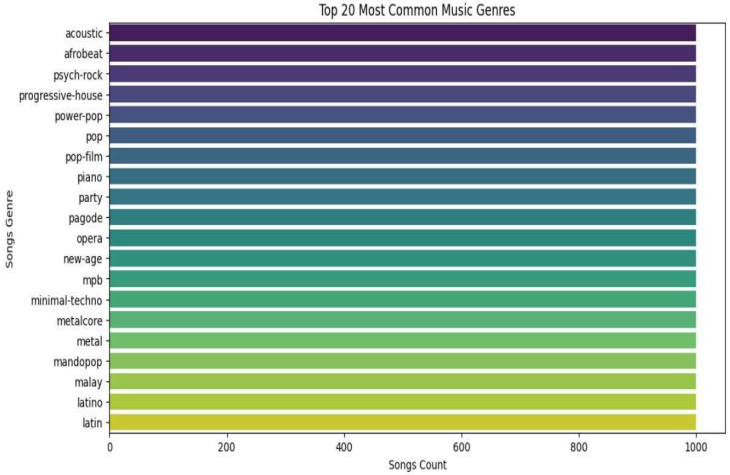


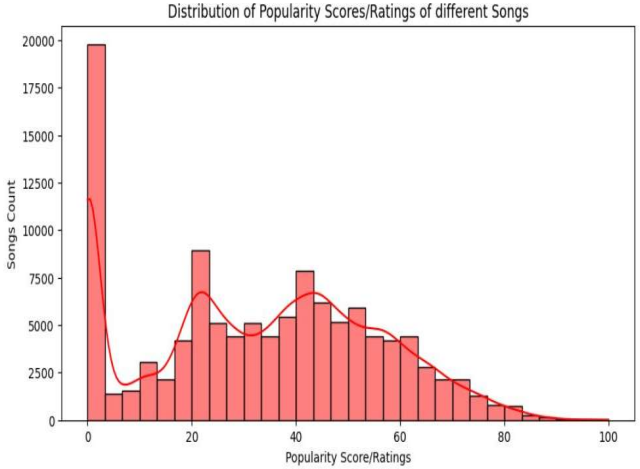


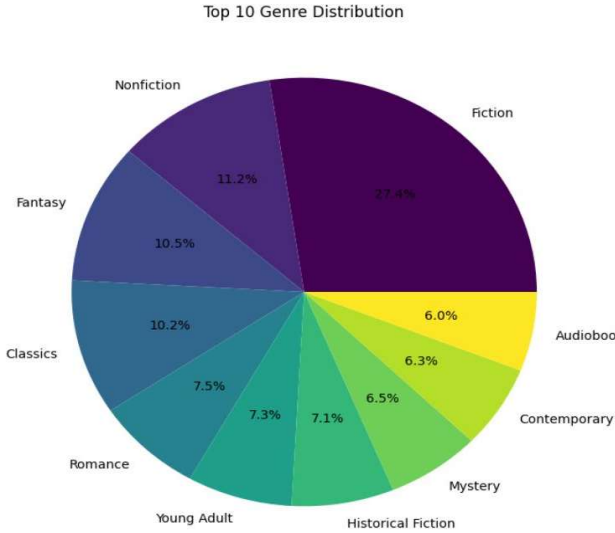
Books Dataset Trends:



Music Dataset Trends:







Milestone 2: Model Selection, Training & Evaluation

Model Selection: GRU4Rec

- **Architecture**: GRU4Rec (Gated Recurrent Units)
  - **Embedding Layer**: Converts item IDs to dense vectors.
  - **GRU Layer**: Captures session-based dependencies.
  - **Fully Connected Layer**: Maps GRU output to item probabilities.
- **Reason**: Suitable for sequential recommendations based on past user interactions.

Model Training:

- **Framework:** PyTorch
- **Training Setup**:
  - **Optimizer**: Adam (lr = 0.001)
  - **Loss**: Cross-Entropy Loss
  - **Epochs**: 100 with early stopping
  - **Batch Size**: 64
- **Evaluation**: Evaluated using **Precision@K**, **Recall@K**, and **F1 Score@K** (k - top k recommendations)

Evaluation metrics & results:

- **Precision@10**: 0.86
- **Recall@10**: 0.89
- **F1 Score@10**: 0.87

```
# Example session (list of previously interacted Item_IDs)
example_session = ["Movie_Creed III", "Movie_Mummies", "Movie_Supercell"]

# Get top 5 recommendations
recommendations = recommend_next_items(example_session, top_k=5)
print("Recommended Items:", recommendations)
```

```
Recommended Items: ['Movie_Ford v Ferrari', 'Music_5awljpWNO5TpXCyjpvCBbs', 'Music_29RiulWABWHcTRLkDqVCl1', 'Music_69Jv0CiMlrpfjh9N2WFkr0', 'Music_40o76
OwDazc0h2QrZhWl']
```

Milestone 3: Model Integration and Deployment

Future Work

- Tune **hyperparameters** of the model for optimal performance(better F1-Score).
- Experiment with **hybrid models** (e.g., combining collaborative filtering.)
- Develop an application using React (frontend) and Flask (backend) to deploy the recommendation system and serve real-time recommendations.
- Use Large Language Models (LLMs) to create more contextually relevant Item_IDs for better cross-domain integration (e.g., mapping movie, music, and book data to a unified representation).