PHASE-III REPORT

|  |  |  |  |
| --- | --- | --- | --- |
| **Karunawansa Yasiru** | **Mehta Akshat** | **Tidwell Kippy** | **Xu Shengyan** |
| **Yik4325** | **Am1717** | **Kct2548** | **Sx6553** |

**Table of Contents**

1. **Introduction**  
   Overview of the Data Cleaning Process  
   Objectives of Frequent Itemset Mining and Association Rule Mining
2. **Data Preparation and Cleaning**  
   Loading the Data  
   Handling Missing Data  
   Data Transformations
3. **Methodology**  
   Frequent Itemset Mining  
   Professions Mining  
   Genres Mining  
   Association Rule Mining  
   Metrics Used for Rule Mining
4. **Code Explanation**  
   Frequent Itemset Mining (mine\_frequent\_itemsets)  
   Association Rule Mining (generate\_rules)
5. **Results and Analysis**  
   Frequent Itemsets for Professions  
   Frequent Itemsets for Genres  
   Frequent Genre Pairs  
   Frequent Profession Pairs
6. **Conclusion**  
   Key Findings and Insights for the Film Industry

**Data Cleaning Process for IMDb Dataset**

The provided Python script will clean multiple IMDb data files by applying specific cleaning steps for each dataset.

**General Idea**

1. **Load and Validate Data:**
   * Use pandas to load TSV files and validate the column structure.
2. **Standardize Missing Values:**
   * Replace /N with None to ensure consistent handling of missing data.
3. **Ensure Data Types:**
   * Convert fields to appropriate types (e.g., integers, floats, booleans, lists) based on their expected structure.
4. **Remove Duplicates:**
   * Remove duplicate rows based on unique identifiers or combinations of key fields.
5. **Handle Invalid Rows:**
   * Skip rows with invalid column counts or irrecoverable issues, ensuring data quality.

**Instructions on how to run the program:**

**Required Tools:** pycharm

Place the original data files in a directory named “data”. The newly created “data” directory should be placed in the same directory as the “clean\_data.py” python script. After running the script successfully, the cleaned data files should be placed in a new directory named “cleaned\_data”.

**1. title.akas.tsv.gz**

**Structure**

* **Fields:**
  + titleId: Unique identifier for the title (tconst).
  + ordering: Row ordering for alternate titles.
  + title: Localized or alternative title.
  + region: Region of the title.
  + language: Language of the title.
  + types: Attributes for the title (e.g., "alternative", "dvd").
  + attributes: Additional attributes for the title.
  + isOriginalTitle: Boolean indicating if it is the original title, 0: not original; 1: original.

**Cleaning Steps**

1. **Ensure Valid Column Count:**
   * Rows must have exactly 8 columns.
   * Rows with invalid column counts are skipped.
2. **Handle Missing Values:**
   * Replace /N with None for region, language, types, and attributes.
3. **Ensure Data Types:**
   * Convert ordering to an integer.
   * Convert isOriginalTitle to a boolean.
4. **Remove Duplicates:**
   * Remove duplicate rows based on titleId and ordering.

**2. title.basics.tsv.gz**

**Structure**

* **Fields:**
  + tconst: Unique identifier for the title.
  + titleType: Type of title (e.g., movie, short).
  + primaryTitle: Most common or promotional title.
  + originalTitle: Title in its original language.
  + isAdult: Boolean indicating if the title is adult content, 0: not original; 1: original.
  + startYear: Release year for TV series.
  + endYear: End year for TV series.
  + runtimeMinutes: Duration in minutes.
  + genres: List of genres associated with title.

**Cleaning Steps**

1. **Ensure Valid Column Count:**
   * Rows must have exactly 9 columns.
   * Rows with invalid column counts are skipped.
2. **Handle Missing Values:**
   * Replace /N with None for endYear, runtimeMinutes, and genres.
3. **Ensure Data Types:**
   * Convert isAdult to a boolean.
   * Convert startYear, endYear, and runtimeMinutes to integers.
   * Split genres into a list of strings.
4. **Remove Duplicates:**
   * Remove duplicate rows based on tconst.
5. **Invalid Data:**
   * Check data types for runtimeMinutes and genres and assign None and an empty array respectively if data types are invalid.

**3. title.principals.tsv.gz**

**Structure**

* **Fields:**
  + tconst: Unique identifier for the title.
  + ordering: Row ordering for principal entries.
  + nconst: Unique identifier for the person.
  + category: Job category (e.g., actor, director).
  + job: Specific job title.
  + characters: Characters played (if applicable).

**Cleaning Steps**

1. **Ensure Valid Column Count:**
   * Rows must have exactly 6 columns.
   * Rows with invalid column counts are skipped.
2. **Handle Missing Values:**
   * Replace /N with None for job and characters.
3. **Ensure Data Types:**
   * Convert ordering to an integer.
   * Attempt to split into a list of characters is unfeasible as field is not generalized, therefore will be stored as a string after removing extra characters from the field.
4. **Remove Duplicates:**
   * Remove duplicate rows based on tconst, ordering, and nconst.

**4. title.ratings.tsv.gz**

**Structure**

* **Fields:**
  + tconst: Unique identifier for the title.
  + averageRating: Weighted average rating.
  + numVotes: Number of votes.

**Cleaning Steps**

1. **Ensure Valid Column Count:**
   * Rows must have exactly 3 columns.
   * Rows with invalid column counts are skipped.
2. **Handle Missing Values:**
   * Assign None if data is not present or invalid.
3. **Ensure Data Types:**
   * Convert averageRating to a float.
   * Convert numVotes to an integer.
4. **Remove Duplicates:**
   * Remove duplicate rows based on tconst.

**5. name.basics.tsv.gz**

**Structure**

* **Fields:**
  + nconst: Unique identifier for the person.
  + primaryName: Person's most often credited name.
  + birthYear: Year of birth.
  + deathYear: Year of death (if applicable).
  + primaryProfession: Top 3 professions as a list.
  + knownForTitles: List of titles (tconst) the person is known for.

**Cleaning Steps**

1. **Ensure Valid Column Count:**
   * Rows must have exactly 6 columns.
   * Rows with invalid column counts are skipped.
2. **Handle Missing Values:**
   * Replace /N with None for birthYear, deathYear, primaryProfession, and knownForTitles.
3. **Ensure Data Types:**
   * Convert birthYear and deathYear to integers.
   * Split primaryProfession and knownForTitles into lists.
4. **Remove Duplicates:**
   * Remove duplicate rows based on nconst.

**Frequent Itemset Mining and Association Rule Mining**

**1. Introduction**

In this project, we applied **frequent itemset mining** and **association rule mining** techniques to discover interesting relationships between genres and artist professions in a movie dataset. Using the **Apriori algorithm**, we aimed to:

1. **Identify professions that frequently co-occur among artists.**
2. **Discover common genres associated with specific titles.**

The project utilized **frequent itemset mining** to identify itemsets (genres or professions) that appear together frequently and **association rule mining** to derive rules that describe relationships between these itemsets. The results are valuable for understanding genre pairings, profession co-occurrence, and uncovering trends within the movie industry.

**2. Data Preparation and Cleaning**

Before applying the **Apriori algorithm**, the data had to be cleaned to ensure accuracy and consistency. This involved the following steps:

1. **Loading the Data**: The cleaned data was loaded into Pandas DataFrames. We worked with two primary sources of data:
   * **Artist professions**: Data from the artist\_profession table (containing artist identifiers and their associated professions).
   * **Title genres**: Data from the title\_genre table (containing movie identifiers and associated genres).
2. **Handling Missing Data**: Missing or invalid values were replaced with None or removed where necessary.
3. **Transformations**:
   * Multi-valued columns (e.g., multiple professions for an artist or multiple genres for a movie) were split into individual entries to prepare them for analysis.

After data cleaning and transformation, we were ready to apply the **Apriori algorithm**.

**3. Methodology**

**3.1. Frequent Itemset Mining**

The first step in the analysis was to identify **frequent itemsets**—sets of items (such as professions or genres) that occur together frequently in the dataset. The **Apriori algorithm** was used to mine these frequent itemsets. Here’s how the process was carried out:

* **Professions Mining**: For artist professions, we treated each artist's set of professions as a transaction and used the **Apriori algorithm** to identify combinations of professions that frequently co-occur among artists.
* **Genres Mining**: For movie genres, we treated each movie’s set of genres as a transaction and used the **Apriori algorithm** to identify genre pairings (e.g., Drama and Romance) that frequently co-occur.

The **Apriori algorithm** works by generating candidate itemsets in a **bottom-up** manner. It first identifies individual frequent items (e.g., genres or professions) and then extends the itemsets to find combinations of items that are frequent across the dataset.

**3.2. Association Rule Mining**

Once we obtained frequent itemsets, the next step was to generate **association rules** from these itemsets. Association rule mining helps us identify the relationships between items in the dataset. For example, we can discover that "if a movie is categorized as **Action**, it is likely to also be categorized as **Adventure**."

* **Metrics Used**:
  + **Support**: Measures the proportion of transactions that contain both the antecedent and the consequent of the rule.
  + **Confidence**: Measures the probability that the consequent occurs given the antecedent.
  + **Lift**: Measures the strength of a rule over the expected chance of the antecedent and consequent co-occurring randomly.

**4. Code Explanation**

**Frequent Itemset Mining (mine\_frequent\_itemsets)**

The **mine\_frequent\_itemsets** function uses the **Apriori algorithm** to find **frequent itemsets** based on the **minimum support** threshold. It works by:

1. **Loading Data**: Loading the prepared transactions data (i.e., one-hot encoded genre or profession data).
2. **Applying Apriori**: Using the apriori() function from mlxtend.frequent\_patterns to identify frequent itemsets.
3. **Support Calculation**: The support of the itemsets is calculated and itemsets that meet the minimum support threshold are considered frequent.

**Association Rule Mining (generate\_rules)**

The **generate\_rules** function takes the **frequent itemsets** as input and generates **association rules** based on the specified **confidence** and **lift** metrics. It works by:

1. **Generating Rules**: Using the association\_rules() function from mlxtend.frequent\_patterns to generate rules.
2. **Evaluating Rules**: Evaluating the rules based on **confidence** and **lift** to find the most meaningful relationships.
3. **Saving Results**: The association rules, including support, confidence, and lift values, are saved for further analysis.

**5. Results and Analysis**

**5.1. Frequent Itemsets for Professions**

The **l1\_profession\_efficient\_apriori** output contains the most frequent professions among artists. Here are the top professions by frequency:

* **Actor**: 3,230,340 occurrences
* **Actress**: 1,952,939 occurrences
* **Producer**: 1,239,724 occurrences
* **Director**: 758,624 occurrences
* **Writer**: 928,597 occurrences

These frequent itemsets highlight the dominant roles in the film industry, with **actors** and **actresses** being the most frequent, followed by **producers**, **writers**, and **directors**.

**5.2. Frequent Itemsets for Genres**

The **l1\_genres\_efficient\_apriori** output contains the most frequent genres in the dataset. Here are the top genres:

* **Drama**: 3,176,217 occurrences
* **Comedy**: 2,191,603 occurrences
* **Romance**: 1,052,275 occurrences
* **Documentary**: 1,069,460 occurrences
* **Action**: 465,079 occurrences

The genre **Drama** appears most frequently, followed by **Comedy** and **Romance**, which are traditionally popular genres in the film industry.

**5.3. Frequent Genre Pairs (l2\_genres\_efficient\_apriori)**

The **l2\_genres\_efficient\_apriori** output shows frequent genre pairings. For example:

* **Action and Adventure**: 165,378 occurrences
* **Action and Drama**: 133,997 occurrences
* **Action and Comedy**: 95,528 occurrences
* **Action and Sci-Fi**: 18,497 occurrences

These results suggest that **Action** is commonly paired with **Adventure**, **Drama**, and **Comedy**. This makes sense, as action films often blend with these genres to appeal to a wider audience.

**5.4. Frequent Profession Pairs (l2\_professions\_efficient\_apriori)**

The **l2\_professions\_efficient\_apriori** output shows frequent profession pairs. For example:

* **Actor and Director**: 127,652 occurrences
* **Actor and Producer**: 154,034 occurrences
* **Actor and Writer**: 183,432 occurrences
* **Actress and Director**: 39,479 occurrences

The results show that **actors** frequently collaborate with **directors** and **producers**. Interestingly, **actors** also often work as **writers**, highlighting the multidimensional roles some individuals play in film production.

**5.5. Frequent Genre Pairs (l3\_genres\_efficient\_apriori)**

The **l3\_genres\_efficient\_apriori** output shows frequent genre pairings. For example:

• **Action, Adventure and Animation**: 77,311 occurrences

• **Action, Crime and Horror**: 43,689 occurrences

• **Adventure, Animation and Comedy**: 61,346 occurrences

• **Animation, Comedy and Family**: 59,190 occurrences

These results represent more specific pairs of film genres that will be more appealing to audiences. For **Action** movie, combining with **Horror** or **Animation** will appeal more audiences.

**5.6. Frequent Profession Pairs (l3\_professions\_efficient\_apriori)**

The **l3\_professions\_efficient\_apriori** output shows frequent profession pairs. For example:

• **Actor, Composer and Music\_department**: 14,905 occurrences

• **Actor, Director and Producer**: 21,141 occurrences

• **Actor, Director and Writer**: 64,807 occurrences

• **Actor, Producer and Writer**: 39,261 occurrences

The result shows that **actor** collaborates mainly with **composer, director, writer** and **producer**, showing the high frequency of these profession work in film production.

**6. Conclusion**

This project successfully applied **frequent itemset mining** and **association rule mining** to uncover meaningful relationships between movie genres and artist professions. Using the **Apriori algorithm**, we identified frequent itemsets and generated association rules to reveal how genres and professions co-occur in the film industry.

**Key Findings:**

1. **Frequent Professions**: The most common professions are **actor**, **actress**, **producer**, and **director**.
2. **Frequent Genre Pairings**: **Action** is commonly paired with **Adventure**, **Comedy**, and **Drama**.
3. **Profession Pairings**: **Actors** are frequently associated with **directors**, **producers**, and **writers**.

These insights provide valuable information about trends in the film industry, which can be used for recommendation systems, market analysis, and predicting future trends in content creation.