ANIME DATA MINING

Data Mining – Semester 01 – 2025/2026

Members

Đỗ Hùng Việt – ITCSIU22197

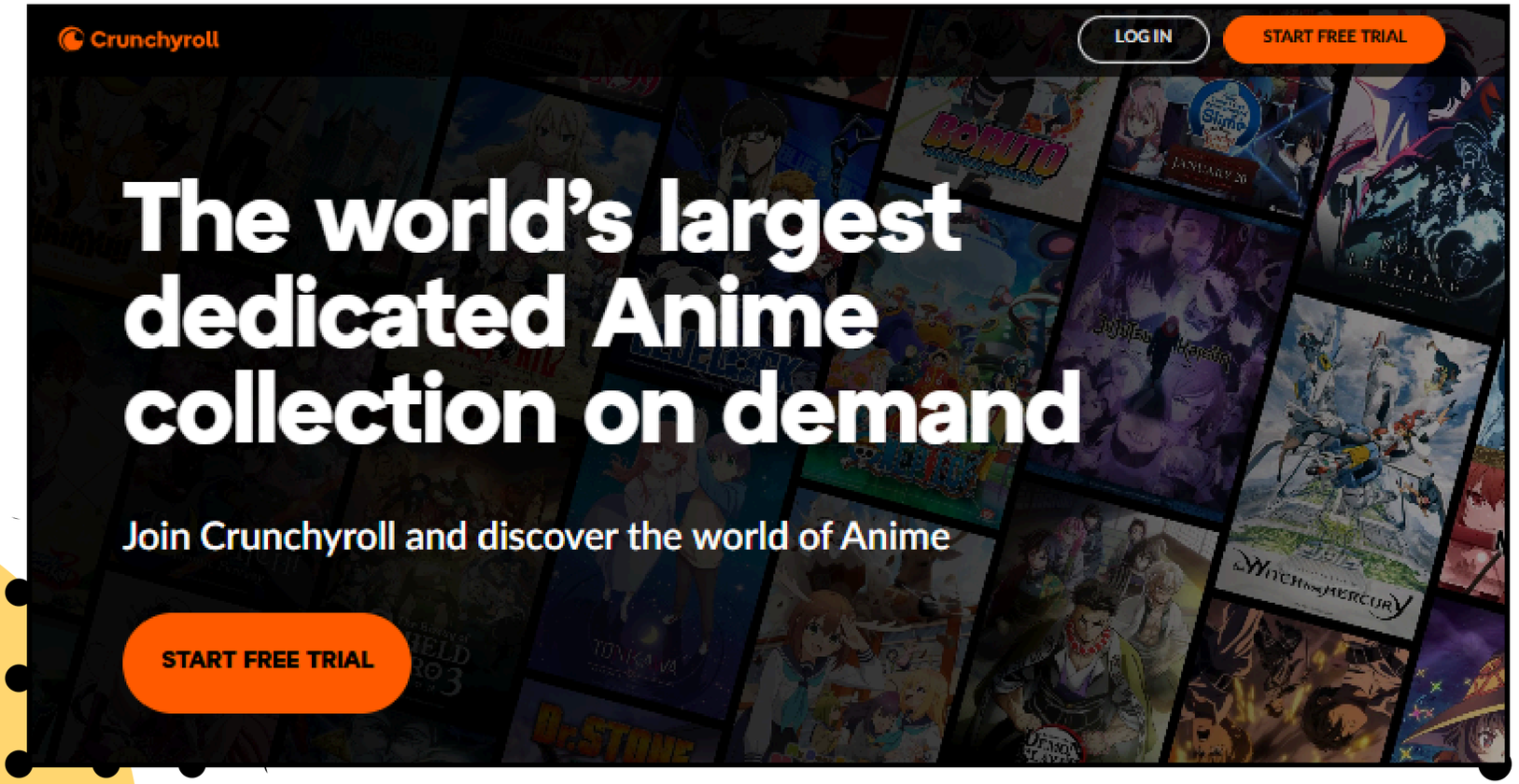
Hà Minh Trí – ITCSIU22194

Outline

1. Introduction
2. Implementations
3. Improvements
4. Evaluation
5. Conclusion

Introduction

Introduction



Frontpage of Crunchyroll

Introduction

Users rating movies

Implementations

Data Preparation and Processing

1. Preparation: Loading & Inspecting Data
2. Processing: Cleaning & Basic Encoding
3. Feature Selection: Merging & Filtering

Preparation

1

Source: Kaggle Anime Recommendation Database

2

- Tools:
  - Python (Pandas) for cleaning.
  - Weka Library (Java) for modeling.

Processing (Data Description)

1

Anime Dataset

- 12,294 instances (anime series).
- Attributes: Genre, Type, Episodes, Members
- Characteristics: Multi-valued attributes (Genre), Missing values ("Unknown")

```
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   anime_id  12294 non-null  int64
 1   name      12294 non-null  object
 2   genre     12232 non-null  object
 3   type      12269 non-null  object
 4   episodes  12294 non-null  object
 5   rating    12064 non-null  float64
 6   members   12294 non-null  int64
dtypes: float64(1), int64(2), object(4)
memory usage: 672.5+ KB
None
```

2

Rating Dataset

- 7.8 Million entries.
- Attributes: User_ID, Anime_ID, Rating (-1 to 10)
- Characteristics: High sparsity, Implicit ratings (-1 represents "watched but not rated")

```
 #   Column    Dtype
---  ------    -----
 0   user_id   int64
 1   anime_id  int64
 2   rating    int64
dtypes: int64(3)
memory usage: 178.8 MB
None
```

Processing (Handling Issues)

1

Genre Handling (Basic)

- Took only the first genre from the list
- Drawback: Loss of information (e.g., "Action, Comedy" becomes just "Action")

2

Handling Missing Values

- Categorical: Replaced with "Unknown"
- Numerical: Filled with Median

3

Binning (Discretization)

- Episodes: Short, Medium, Long, Very_Long_Series
- Members: Low, Medium, High

4

Encoding Ratings

- Converted 1–10 scale into: Low (<6), Average (6–8), High (⩾8)
- Merged "–1" into Low (Initial mistake)

Model Development

1. ZeroR (Baseline)
2. OneR (Single Rule)
3. Naive Bayes (Probabilistic)
4. J48 (Decision Tree)

Model Comparison

1 OneR

- Uses one single attribute (e.g., "Type")
- Simple baseline rule
- Fast but low accuracy on complex data

2 Naive Bayes

- Uses all attributes independently
- Good for text / categorical data
- Scalable to large datasets

3 J48 (Decision Tree)

- Builds a tree by splitting attributes (Entropy /Information Gain)
- Captures interactions between Genre & Episodes
- Highest complexity but interpretable

Results

Metrics	ZeroR	OneR	Naïve Bayes	J48
Correctly Classified Instances	53.79 %	65.87 %	69.11 %	72.34 %
Incorrectly Classified Instances	46.21 %	34.13 %	30.89 %	27.66 %
Kappa statistic	0	0.3276	0.4027	0.4596
Mean absolute error	0.3651	0.2275	0.2509	0.2505
Root mean squared error	0.4272	0.477	0.3746	0.3565
Relative absolute error	100 %	62.3203 %	68.7246 %	68.6227 %
Root relative squared error	100 %	111.6466 %	87.6664 %	83.4502 %

Table 4. Initial Evaluation

Improvements

Dataset Processing Improvement

1 Genre Clustering (TF-IDF)

- Problem: Taking the 1st genre lost data
- Solution: Used TF-IDF Vectorizer to convert genre strings to numbers
- Applied K-Means Clustering (k=10) to group similar anime (e.g., "Action-Adventure" cluster)

2 Advanced Binning

- Members: Split into 8 distinct clusters based on popularity distribution
- Episodes: Categorized strictly (Movie/Special vs Series length)

3 Handling Implicit Ratings

- Problem: "-1" distorted the "Low" rating class
- Solution: Encoded "-1" as a separate class "0" (No_Rating) to preserve pattern

Model Improvement

1 Input Feature Refinement

- Previous: Raw strings (Genre) & Simple Bins
- Improved: Input features are now Cluster IDs (from K-Means) and Encoded Integers
- Benefit: Reduces noise and dimensionality, allowing classifiers (especially J48) to build more efficient trees

2 Target Class Redefinition

- Previous: 3 Classes (Low, Avg, High) – implicitly merged "-1" into "Low"
- Improved: 6 Classes (0 to 5)
- 0: No Rating (Explicitly separated "-1")
- 1-5: User Rating Scale.
- Goal: Helps the model distinguish between "Dislike" vs "Not Rated"

3 Evaluation Protocol (Weka)

- Method: 10-Fold Cross-Validation
- Focus: Shifted from pure Accuracy to Kappa Statistic & TP Rate (Recall)
- Reason: Accuracy is misleading due to class imbalance; Kappa measures true agreement

Results

Before Improvement

- Genre: Only 1st keyword
- User Rating: -1 mixed with Low
- Result: High accuracy on the Majority class, 0% on the Minority class.

After Improvement

- Genre: TF-IDF Clusters (More detailed)
- User Rating: Separated "-1"
- Result: Balanced performance
  - J48 Accuracy: ~70.26% (More realistic)
  - Kappa: Increased to 0.53 (Better agreement)
  - MAE: Decreased to 0.15 (Lower error)

Metrics	ZeroR	OneR	Naïve Bayes	J48
Correctly Classified Instances	46.45 %	61.38 %	63.91 %	70.26 %
Incorrectly Classified Instances	53.55 %	38.62 %	36.09 %	29.74 %
Kappa statistic	0	0.34	0.4572	0.5318
Mean absolute error	0.2712	0.1545	0.1718	0.1552
Root mean squared error	0.3682	0.393	0.3021	0.2823
Relative absolute error	100 %	56.9591 %	63.3415 %	57.23 %
Root relative squared error	100 %	106.7378 %	82.0365 %	76.677 %

Table 5. Advanced Evaluation

Model Evaluation

Model	Accuracy	Kappa	MAE	RMSE	Weighted Avg. F1
OneR	61.38%	0.34	0.1545	0.393	0.614
Naive Bayes	63.91%	0.4572	0.1718	0.3021	0.618
J48	70.26%	0.5318	0.1552	0.2823	0.680

Table 7. Advanced Performance Metrics

- J48: The best overall. Handles the interaction between "Genre Cluster" and "Popularity" effectively.
- Naive Bayes: Having good stability and fast execution speed.

Analysis

1

Trade-offs

- J48: Most accurate & interpretable but requires memory/time for tree building.
- Naive Bayes: Faster but assumes independence (weakness with correlated genre clusters).

2

Challenges

- Class Imbalance: "High" ratings still dominate.
- Noise: Multi-genre animes are hard to classify perfectly even with clustering.
- Scalability: 7.8M ratings required sampling (10k instances) for Weka to run.

Conclusion

Key Findings

- Preprocessing is Key: TF-IDF + K-Means clustering significantly reduced noise compared to simple string splitting.
- Data Quality: Separating implicit ratings (-1) improved model reliability.
- Model Selection: Tree-based models (J48) work best for this structured, categorical dataset.

Future Works

- Class Balancing: Apply SMOTE to generate synthetic samples for rare classes (e.g., Very Long Series).
- Deep Learning: Use NLP models (BERT) on anime names/synopses.
- Deployment: Build a Web App recommendation system.

Thank you