Anime Recommendation Model

Luis Parra
ChecoParraLuis@gmail.com

***Abstract*— This project aims to develop a recommendation system for anime shows based on user preferences. The approach involved joining an Anime Identification and User Ratings datasets, performing exploratory data analysis (EDA), cleaning the data, and building a Random Forest classifier. The model was evaluated using accuracy and ROC-AUC scores. The final section outlines the top 5 anime recommendations for specific users.**

1 EXPLORATORY ANALYSIS (EDA)

The EDA phase focuses on understanding the datasets and gaining insight into the distribution of key variables. A description of each variable can be found in the appendix. Some key points covered in this section include:

- Dataset Overview: Datatypes and Joining on common variables.
- Genre Exploration: Analyzing top genres

1.1 Dataset Overview

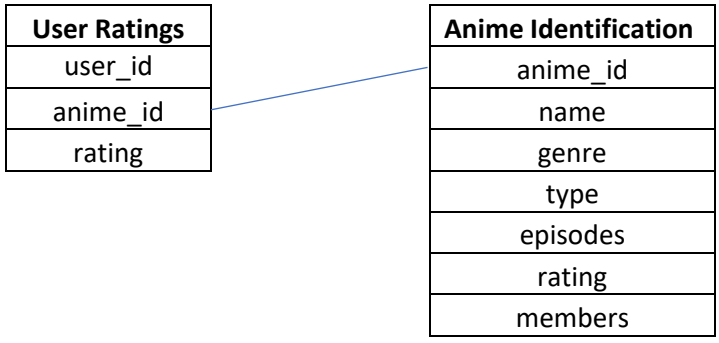

Fig 1.1.1 Datasets Variables

The Anime Identification (anime_df) and the User Ratings table (animeratings_df) are joined on the anime_id variable. This variable is unique in the anime_df therefore I would be doing a left join on the animeratings_df so that I can output all the rows from the users and only the rows that are common in

the anime_df. In mathematics this is referred to surjective if all datapoints in the anime_df are being targeted or a many-to-one relationship. After left-joining these datasets I end up with the merged_anime_ratings dataframe which I will be using for feature engineering and creating the classification model.

Merged Anime Ratings	Dtypes
user_id	int64
anime_id	int64
rating_x	int64
name	object
genre	object
type	object
episodes	object
rating_y	float64
members	float64

Fig 1.1.2 merged_anime_rating dataframe

1.2 Genre Exploration

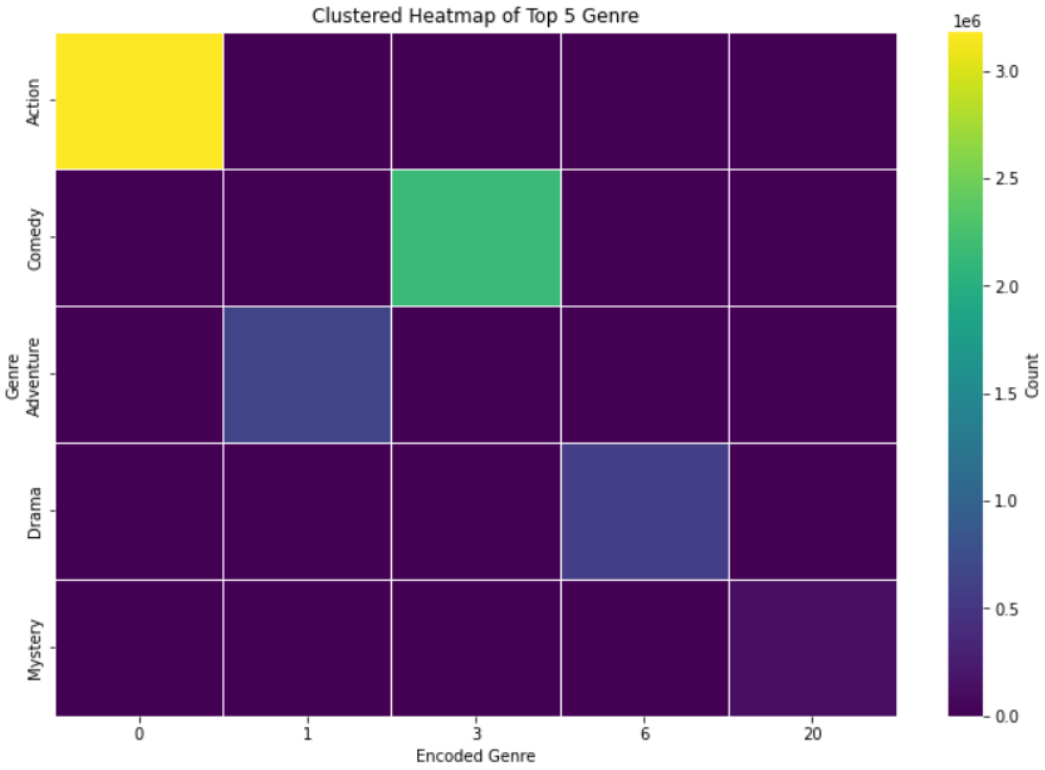

Fig 1.2.1 Heatmap of Genres

This heat map shows the concentration of the top 5 genres using a technique called label encoding. As the variable is categorical label encoding assigns each genre a unique number. The scale is scaled down for better representation of the groups. The y-axis is in the 100,000 units.

2 DATA CLEANING

Data cleaning was a crucial step to ensure the quality and integrity of the dataset. Key steps included:

- Feature Engineering: Creating new columns for the top two genres for each row. Creating binary column for rating_x.
- Handling Missing Values: Dropping rows with missing values.
- Numeric Conversion: Converting non-numeric data to numeric for modeling purposes.

2.1 Feature Engineering

First, I have to manipulate variables for modeling purposes. As the model needs a binary target variable, I chose the users rating (rating_x). I set a threshold where any ratings over 8.0 were considered True and the rest are False. Also, considering the genre variable was a list of genres for each row, I also separated this out by create two new columns, genre1 and genre2, with the first two genres from the genre column.

	user_id	anime_id	rating_x	name	genre	type	episodes	rating_y	members	liked	genre1	genre2
0	1	20	-1	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297.0	False	Action	Comedy
1	1	24	-1	School Rumble	Comedy, Romance, School, Shounen	TV	26	8.06	178553.0	False	Comedy	Romance
2	1	79	-1	Shuffle!	Comedy, Drama, Ecchi, Fantasy, Harem, Magic, R...	TV	24	7.31	158772.0	False	Comedy	Drama
3	1	226	-1	Elfen Lied	Action, Drama, Horror, Psychological, Romance,...	TV	13	7.85	623511.0	False	Action	Drama
4	1	241	-1	Girls Bravo: First Season	Comedy, Ecchi, Fantasy, Harem, Romance, School	TV	11	6.69	84395.0	False	Comedy	Ecchi

Fig 2.1.1 New merged_anime_rating df with additional columns

2.2 Handling Missing Values

My method of handling Missing values was to eliminate those rows from the dataset. Since there are about 8 million rows in this dataset eliminating 200,000 rows should have little impact to the model. Given that this is a project is sufficient for the scope and purpose of this project. There are other ways of handling missing data such as imputing with averages or assigning the missing group of data as “others”.

2.3 Numeric Conversion

The predicted variable (liked) can be left as binary values. To handle the categorical data I used the method of One-Hot Encoding which creates a linear dataframe of 0's and 1's to make each value in the category unique.

3 MODEL BUILDING

The model building comprises of feature selection, splitting the data, and training a Random Forest classifier. Highlights include:

- Feature Selection: Choosing relevant features for the model.
- Model Training: Utilizing a Random Forest classifier for binary classification.

3.1 Feature Selection

The target variable will be the "liked" column that was created. As the purpose of this model is to predict whether a user will like an anime show this should be binary in nature. The predictor variables I chose are genre1, genre2, type, episodes, rating_y and members.

```
# Features and target variable
X = merged_anime_ratings[['genre1', 'genre2', 'type', 'episodes', 'rating_y', 'members']]
y = merged_anime_ratings['liked']
```

Fig 3.1.1 Python code of Feature Selection

3.2 Random Forest Classification

With the X and y variables declared I am able to split and train the data. A good rule of thumb is have a test set of 30% to 5%. For this project used 10% as I wanted the model to be trained on a large training set. I also set the number of trees in the classifier to 30. This number can range depending on computing power.

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)

# Create a Random Forest Classifier
rf = RandomForestClassifier(n_estimators=30, random_state=42)

# Train the model
rf.fit(X_train, y_train)

# Make predictions on the test set
y_pred = rf.predict(X_test)
```

Fig 3.2.1 Python code of Model

4 RESULTS

The model demonstrated promising results in predicting whether a user would like an anime show based on selected features. Metrics used are:

Accuracy: Achieving an accuracy score of [0.628].

ROC-AUC Score: Attaining a ROC-AUC score of [.676].

Confusion Matrix: Visualizing model performance through a confusion matrix.

4.1 Accuracy

Accuracy is derived from the confusion matrix. This is the total number of correct guess over all guesses.

4.2 ROC-AUC Score

ROC - AUC measures the models ability to distinguish between classes. A score closer to 1 yields better classification results.

4.3 Confusion Matrix

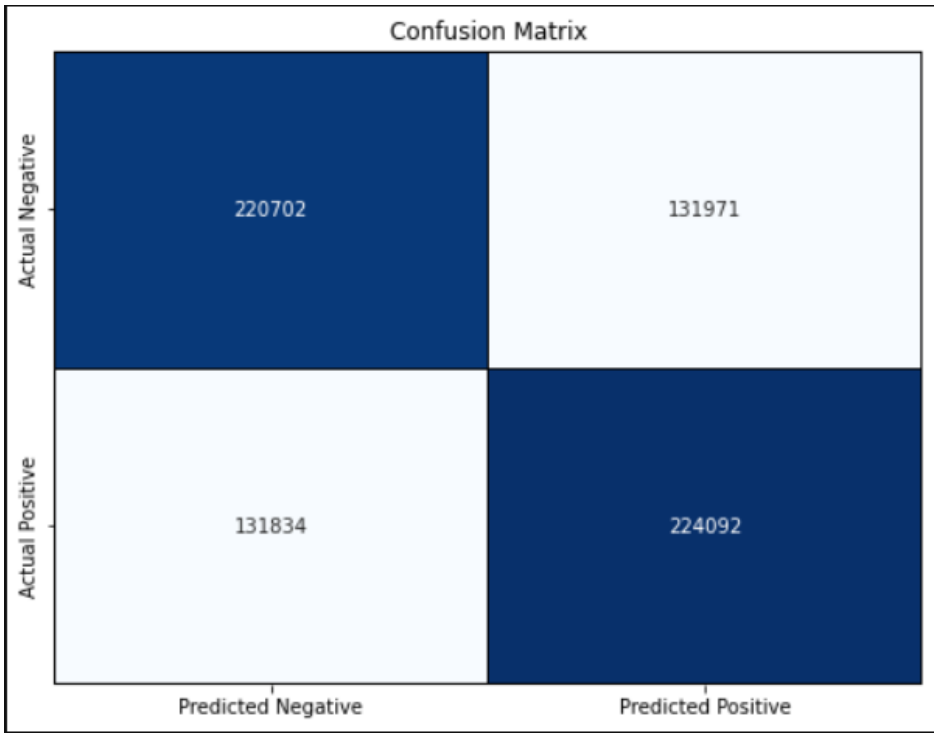

Fig 4.3.1 Confusion Matrix

The confusion matrix can derive matrix such as accuracy, precision and F1 score.

5 ANIME RECOMMENDATIONS

The Random Forest classifier is used to predict the likelihood of a user liking an anime, and this prediction is then used to recommend the top 5 animes to each user. It's a personalized recommendation system that takes into account individual user preferences based on learned patterns from the training data.

Top 5 Anime Recommendations for User 1	
anime_id	**name**
20	Naruto
10080	Kami nomi zo Shiru Sekai II
24	School Rumble
23847 Yahari Ore no Seishun	Love Comedy wa Machigatt...
10793	Guilty Crown

Fig 5.1.1 Anime Recommendations for User 1

```python
# New column for model prediction
merged_anime_ratings['user_liked'] = rf.predict(X)

# Empty dictionary
user_recommendations = {}

# Iterate through each user and recommend top 5 animes
for user_id, group in merged_anime_ratings.groupby('user_id'):
    user_animes = group[['anime_id', 'name', 'user_liked']]
    user_top_5_animes = user_animes.sort_values(by='user_liked', ascending=False).head(5)
    user_recommendations[user_id] = user_top_5_animes[['anime_id', 'name']]

# Declare user and check
specific_user_id = 1
print(f"Top 5 Anime Recommendations for User {specific_user_id}:")
print(user_recommendations[specific_user_id])
```

Fig 5.1.2 Python code for predicting on the users target variables

6 CONCLUSION & FUTURE PROGRESS

In conclusion, the project successfully developed a recommendation system for anime shows. The combination of data cleaning, and Random Forest modeling contributed to a creating a quick and effective model. The personalized recommendations offer a valuable feature for enhancing user engagement.

6.1 Future Work

Potential avenues for future work include:

Fine-Tuning the Model: Experimenting with parameter tuning to improve model performance.

User Interface: Creating a user-friendly interface for users to interact with the recommendation system.

Different Approach: Create a content based or collaborative filtering approach.

7 REFERENCES

1. Datasets
    a. https://www.kaggle.com/code/hasibalmuzdadid/anime-ratings-analysis-recommender-system/input

8 APPENDICES

1. Variable Descriptions
    a. user_id: User's identification
    b. anime_id: unique id for anime
    c. rating_x: the user's rating
    d. name: name of anime

e. genre: list of genre Example: action,comedy,drama
f. type: tv, movies, etc
g. episodes: how many episodes in this show. (1 if movie)
h. rating_y: The community members ratings
i. members: total number of member in the community that watched this show
j. genre1: first genre in the list of genre
k. genre2: second genrein the list of genre
l. liked: Binary column where if rating_x is greater than 8.0 then True else False