Pratilipis Recommendation System Documentation

1. Data Preparation and Preprocessing:

1.1 Introduction

This data preprocessing pipeline prepares **user interaction data** and **story metadata** for a recommendation system. The process includes **merging datasets, handling missing or incorrect data, encoding categorical variables, extracting useful features, and normalizing values** to ensure clean and structured data for accurate model predictions.

Dataset (First Five Rows): User Interaction data and Meta data

	user_id	pratilipi_id	read_percent	updated_at
0	5506791961876448	1377786228262109	100.0	2022-03-22 10:29:57.291
1	5506791971543560	1377786223038206	40.0	2022-03-19 13:49:25.660
2	5506791996468218	1377786227025240	100.0	2022-03-21 17:28:47.288
3	5506791978752866	1377786222398208	65.0	2022-03-21 07:39:25.183
4	5506791978962946	1377786228157051	100.0	2022-03-22 17:32:44.777

	author_id	pratilipi_id	category_name	reading_time	updated_at	published_at
0	-3418949279741297	1025741862639304	translation	0	2020-08-19 15:26:13	2016-09-30 10:37:04
1	-2270332351871840	1377786215601277	translation	171	2021-01-21 16:27:07	2018-06-11 13:17:48
2	-2270332352037261	1377786215601962	translation	92	2020-09-29 12:33:57	2018-06-12 04:19:12
3	-2270332352521845	1377786215640994	translation	0	2019-10-17 09:03:37	2019-09-26 14:58:53
4	-2270332349665658	1377786215931338	translation	47	2020-05-05 11:33:41	2018-11-25 12:28:23

1.2 Data Merging (Inner Join)

Why Merge?

We have two datasets:

1. **User Interaction Data** → Contains information about users reading stories (pratilipi_id).
2. **Story Metadata** → Contains information about stories (pratilipi_id).

Since both datasets share (pratilipi_id), an **inner join** is used to merge them.

Why Use an Inner Join?

- It **keeps only the stories that exist in both datasets**.
- If a pratilipi_id exists in user_interaction.csv but **not in metadata.csv**, it means that the story details are missing or invalid.
- If a pratilipi_id exists in metadata.csv but **not in user_interaction.csv**, it means the story was **never read**, so it is irrelevant for user behavior analysis.

Tabular Visualization (After Merge)

	user_id	pratilipi_id	read_percent	interaction_time	author_id	category_name	reading_time	meta_updated_at	published_at
0	5506791961876448	1377786228262109	100.0	2022-03-22 10:29:57.291	-2270332349684758	novels	376	2022-03-15 18:39:52	2022-03-15 18:39:52
1	5506791961876448	1377786228262109	100.0	2022-03-22 10:29:57.291	-2270332349684758	family	376	2022-03-15 18:39:52	2022-03-15 18:39:52
2	5506791961876448	1377786228262109	100.0	2022-03-22 10:29:57.291	-2270332349684758	romance	376	2022-03-15 18:39:52	2022-03-15 18:39:52
3	5506791966456696	1377786228262109	100.0	2022-03-21 07:11:52.070	-2270332349684758	novels	376	2022-03-15 18:39:52	2022-03-15 18:39:52
4	5506791966456696	1377786228262109	100.0	2022-03-21 07:11:52.070	-2270332349684758	family	376	2022-03-15 18:39:52	2022-03-15 18:39:52

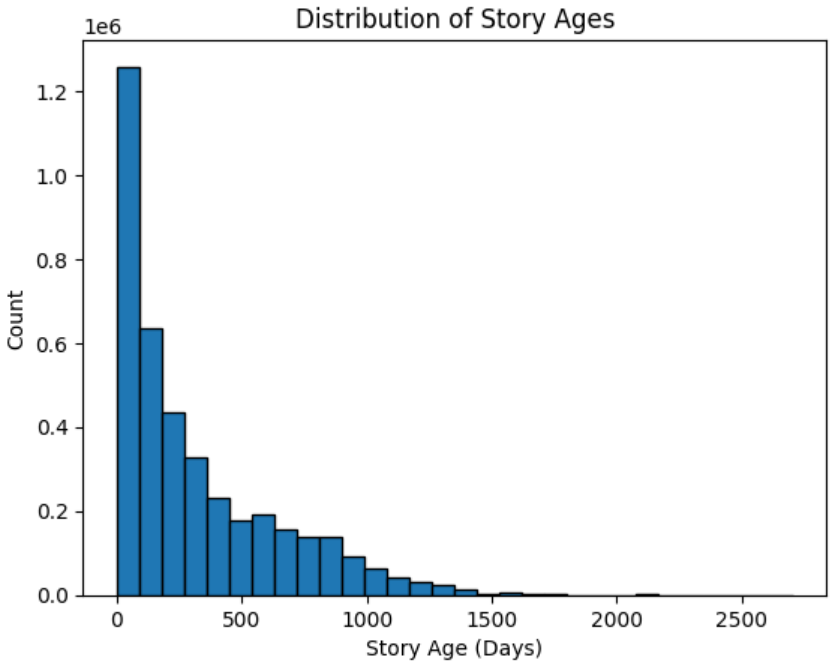
1.3 Feature Engineering

Calculating Story Age (Days)

Formula:

{story age days} = (published at) - (interaction time}

- **Why?** Story age affects reading patterns. Older stories may have higher/lower engagement.
- **What's different?** A newer story may be trending, while an older story may have an established audience.



Removing Invalid Story Age (< 0)

If story_age_days < 0, it means the **interaction happened before the story was published** → Invalid Data.

- **105 Rows Removed**

These were errors where **interaction_time was before published_at**.

Example of Wrong Data (Tabular Representation)

read_percent	interaction_time	author_id	category_name	reading_time	meta_updated_at	published_at	story_age_days	story_age_years
0.0	2022-03-20 09:46:36.062	-2270332330951340	romance	684	2022-03-20 09:47:43	2022-03-20 09:47:43	-1	-1
0.0	2022-03-20 09:46:36.062	-2270332330951340	suspense	684	2022-03-20 09:47:43	2022-03-20 09:47:43	-1	-1
0.0	2022-03-19 08:10:44.656	-2270332349516917	novels	731	2022-03-22 06:30:11	2022-03-22 06:30:11	-3	-1
0.0	2022-03-19 08:10:44.656	-2270332349516917	social	731	2022-03-22 06:30:11	2022-03-22 06:30:11	-3	-1
0.0	2022-03-19 08:10:44.656	-2270332349516917	women	731	2022-03-22 06:30:11	2022-03-22 06:30:11	-3	-1
...	...	...	...	...	...	...	...	...

- These rows were **removed** from the dataset.

1.4 Checked Missing Values

We checked for missing values in the merged dataset using:

Summary

No missing values found in the dataset, ensuring that all columns are complete and ready for further processing.

1.5 Extracting Time-Based Features

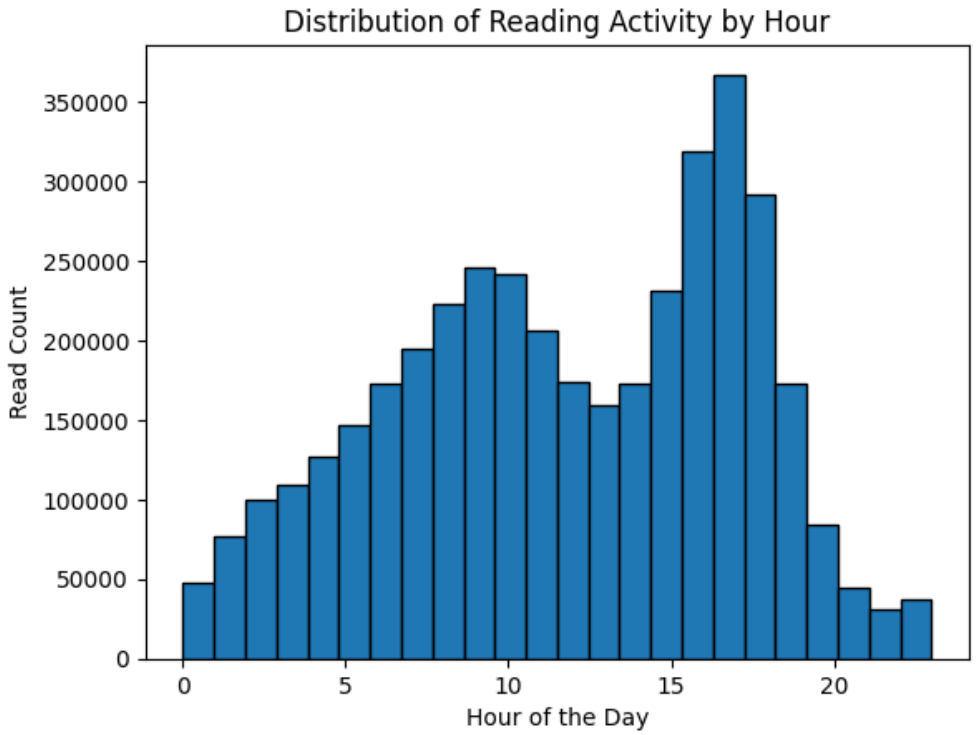
Understanding **when** a user interacts with stories may help with behavioral patterns.

Extracted Features

Feature	Explanation
interaction_hour	What time of the day was the story read? (e.g., night vs. day)
interaction_day	What day of the month? (1 to 31)
interaction_month	Which month? (1 to 12)
interaction_weekday	Which weekday? (Monday to Sunday) – (0 to 6)

- These features help determine if users have **daily/weekly reading habits**.

Graphical Visualization (Time-Based User Behavior)



1.6 Handling Categorical Variables

One-Hot Encoding for category_name

Unique category names:

['novels' 'family' 'romance' 'suspense' 'action-and-adventure' 'webseries'
'swahindi2' 'entertainment' 'women' 'horror' 'social' 'shortstories'
'drama' 'science-fiction' 'Horror-Marathon' 'Pratilipi-kalamkar-samman'
'fantasy' 'crime' 'moral-inspiring' 'experiences-and-memories' 'children'
'life' 'relegion-and-spiritual' 'mythology' 'Pratilipi-Awards-Hindi'
'comedy' 'translation' 'krishi-jeevan' 'short-story-challenge'
'Indiawale' 'Serieswriting' 'erotica' 'detective' 'murder-mystery'
'premkamahina' 'crime-lekhan' 'politics' 'health-and-wellness'
'pratilipi-kids' 'The-Chat-Story' 'Parytan' 'cyber-crime-fiction'
'Rashtriya' 'pravasi-sahitya' 'Radio-Fiction']

Since stories belong to different **categories**, we use **One-Hot Encoding**:

- Creates **binary columns** for each category.
- Prevents **wrong numerical interpretations** (e.g., Romance ≠ 1, Thriller ≠ 2).

Tabular Representation (After One-Hot Encoding)

	pratilipi_id	category_name
0	1377786228262109	novels
1	1377786228262109	family
2	1377786228262109	romance
3	1377786228262109	novels
4	1377786228262109	family

category_name_romance	category_name_science-fiction	category_name_short-story-challenge	category_name_shortstories	category_name_social	category_name_suspense
0	0	0	0	0	0
0	0	0	0	0	0
1	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Label Encoding for user_id, pratilipi_id, author_id:

- Converts **IDs into unique numbers** (e.g., user_id 5506791961876448 → 1, 5506791971543560 → 2).
- Reduces memory usage and **improves efficiency in ML models**.

1.7 Handling Read Percentage Anomalies

Problem:

The read_percent value **should be between 0 and 100**, but **45 rows** had values **greater than 100**.

45 Rows Removed

Example of Wrong Data

	user_id	pratilipi_id	read_percent	interaction_time	author_id	reading_time	meta_updated_at	published_at	story_age_days	story_age_years	...
111306	151272	1443	150.000000	2022-03-22 03:18:00.379	930	840	2020-09-03 14:57:44	2018-07-21 11:55:55	1339	3	...
111307	151272	1443	150.000000	2022-03-22 03:18:00.379	930	840	2020-09-03 14:57:44	2018-07-21 11:55:55	1339	3	...
111308	151272	1443	150.000000	2022-03-22 03:18:00.379	930	840	2020-09-03 14:57:44	2018-07-21 11:55:55	1339	3	...
180364	172071	2584	109.090909	2022-03-20 16:40:53.252	161	15354	2020-11-17 11:25:04	2018-09-12 09:09:43	1285	3	...
180365	172071	2584	109.090909	2022-03-20 16:40:53.252	161	15354	2020-11-17 11:25:04	2018-09-12 09:09:43	1285	3	...
180366	172071	2584	109.090909	2022-03-20 16:40:53.252	161	15354	2020-11-17 11:25:04	2018-09-12 09:09:43	1285	3	...

- These rows were **removed** before modeling.

1.8 Calculating Total and Effective Read Words

Total Read Words

Formula:

total_read_words =(reading_time / 60) x 200

- Assumption: **200 words per minute** reading speed.

Effective Read Words

Formula:

effective_read_words = total_read_words x (read_percent / 100)

- Adjusts **total words** based on **how much of the story was read**.

Difference?

- **Total Read Words** = Maximum words a user **could** have read.
- **Effective Read Words** = Words the user **actually** read.

Tabular Visualization

	read_percent	reading_time	total_read_words	effective_read_words
0	100.000000	376	1253.333333	1253.333333
71839	99.987300	257	856.666667	856.557870
189426	99.986916	362	1206.666667	1206.508786
449936	99.987350	742	2473.333333	2473.020457
583248	99.982740	374	1246.666667	1246.451492

1.9 Feature Normalization (MinMax Scaling & Log Transformation)

MinMax Scaling (0 to 1)

- Applied to **continuous features**:
    - reading_time, story_age_days, interaction_hour, interaction_day, interaction_month, interaction_weekay

	reading_time	story_age_days	story_age_years	interaction_hour	interaction_day	interaction_month	interaction_weekday
0	0.005002	0.002218	0.0	0.434783	0.8	0.0	0.166667
1	0.005002	0.002218	0.0	0.434783	0.8	0.0	0.166667
2	0.005002	0.002218	0.0	0.434783	0.8	0.0	0.166667
3	0.005002	0.001848	0.0	0.304348	0.6	0.0	0.000000
4	0.005002	0.001848	0.0	0.304348	0.6	0.0	0.000000

Log Transformation for effective_read_words

- **Why?**
    - Log transformation **reduces skewness** (some stories have extremely high word counts).
    - Prevents **outliers from dominating the model**.

	effective_read_words
0	7.13436
1	7.13436
2	7.13436
3	7.13436
4	7.13436

1.10 Summary of Data Cleaning Steps

Step	Action Taken	Rows Removed
Merge Datasets	Used **inner join** on pratilipi_id	0
Story Age Calculation	Removed **105 rows** where story_age_days < 0	106
Read Percentage Check	Removed **45 rows** with read_percent < 0 or >100	45
One-Hot Encoding	Converted category_name into binary columns	0
Label Encoding	Converted user_id, pratilipi_id, author_id into numbers	0
Feature Scaling	Used **MinMaxScaler** for normalization	0
Log Transformation	Applied to effective_read_words to reduce skewness	0

- **Errors were identified and removed (Story Age < 0, Read Percent Outliers).**
- **Features were extracted (Story age days, Interaction Hour, Day, Weekday).**
- **Encodings and transformations were applied to make data ML-ready.**
- **Normalization & Log Transformation ensured stability in training.**

1.11 Before training the model, the dataset undergoes several preprocessing steps:

1. Feature Selection

The following features are selected as input for the regression model:

- **User-related features:** user id, author id
- **Temporal features:** story age days, interaction hour, interaction day, interaction month, interaction weekday
- **Categorical features:** One-hot encoded category names (e.g., categoryname romance, category_name social)

The target variable for prediction is:

- **Effective read words** (transformed using log transformation for normalization)

2. Splitting Data

The dataset is split into training and testing sets:

- **Training set:** 75% of the data
- **Testing set:** 25% of the data

Convert Data to PyTorch Tensors

Converting data to PyTorch tensors ensures compatibility with deep learning models, enabling efficient computation, GPU acceleration, and seamless integration with Data loader for optimized training. It also allows automatic differentiation for backpropagation, improving performance in deep learning workflows.

- **batch_size = 128 (for efficient training)**

---

3. Neural Network Model

3.1 Model Architecture

The regression model is a **4-layer fully connected neural network** implemented in PyTorch. The architecture is as follows:

Link: Model Architecture

```
RegressionNN(
  (fc1): Linear(in_features=52, out_features=256, bias=True)
  (batch_norm1): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (dropout1): Dropout(p=0.4, inplace=False)
  (fc2): Linear(in_features=256, out_features=128, bias=True)
  (batch_norm2): BatchNorm1d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (dropout2): Dropout(p=0.4, inplace=False)
  (fc3): Linear(in_features=128, out_features=64, bias=True)
  (batch_norm3): BatchNorm1d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (dropout3): Dropout(p=0.4, inplace=False)
  (fc4): Linear(in_features=64, out_features=1, bias=True)
  (leaky_relu): LeakyReLU(negative_slope=0.01)
)
```

1. **Input Layer:** Accepts the feature vector of size **input_dim** (number of selected features)

2. **Hidden Layers:**
    - **Layer 1:** 256 neurons, Batch Normalization, LeakyReLU activation, 40% dropout
    - **Layer 2:** 128 neurons, Batch Normalization, LeakyReLU activation, 40% dropout
    - **Layer 3:** 64 neurons, Batch Normalization, LeakyReLU activation, 40% dropout
3. **Output Layer:** A single neuron for predicting effective_read_words (continuous regression output)

3.2 Weight Initialization

Weights are initialized using **Xavier Initialization** to improve training stability.

3.3 Optimizer and Loss Function

- **Optimizer:** AdamW with a learning rate of 0.0005 and weight decay 1e-4
- **Loss Function:** Mean Squared Error (MSE)
- **Learning Rate Scheduler:** Step decay reducing LR by 50% every 5 epochs

---

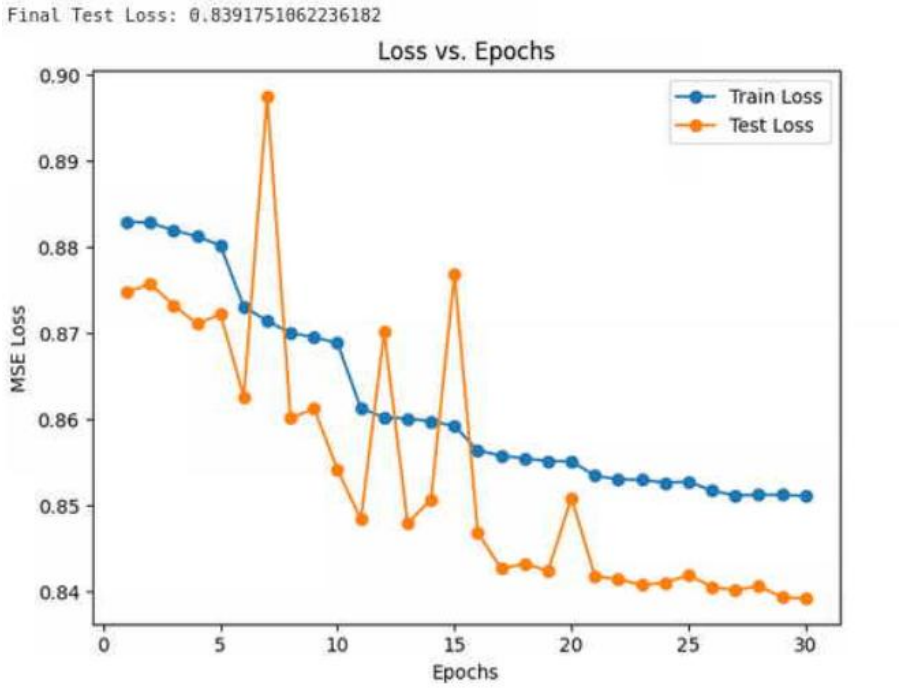
4. Training Process

The model is trained for 30 epochs with early stopping based on validation loss**.**

4.1 Training Logs

```
Epoch 1/30, Train Loss: 0.8830, Test Loss: 0.8748
Epoch 2/30, Train Loss: 0.8828, Test Loss: 0.8757
Epoch 3/30, Train Loss: 0.8820, Test Loss: 0.8733
Epoch 4/30, Train Loss: 0.8813, Test Loss: 0.8711
Epoch 5/30, Train Loss: 0.8802, Test Loss: 0.8722
Epoch 6/30, Train Loss: 0.8731, Test Loss: 0.8626
Epoch 7/30, Train Loss: 0.8714, Test Loss: 0.8976
Epoch 8/30, Train Loss: 0.8700, Test Loss: 0.8601
Epoch 9/30, Train Loss: 0.8696, Test Loss: 0.8613
Epoch 10/30, Train Loss: 0.8689, Test Loss: 0.8542
Epoch 11/30, Train Loss: 0.8613, Test Loss: 0.8484
Epoch 12/30, Train Loss: 0.8602, Test Loss: 0.8701
Epoch 13/30, Train Loss: 0.8601, Test Loss: 0.8479
Epoch 14/30, Train Loss: 0.8598, Test Loss: 0.8507
Epoch 15/30, Train Loss: 0.8593, Test Loss: 0.8769
Epoch 16/30, Train Loss: 0.8564, Test Loss: 0.8468
Epoch 17/30, Train Loss: 0.8558, Test Loss: 0.8427
Epoch 18/30, Train Loss: 0.8555, Test Loss: 0.8433
Epoch 19/30, Train Loss: 0.8552, Test Loss: 0.8424
Epoch 20/30, Train Loss: 0.8551, Test Loss: 0.8509
Epoch 21/30, Train Loss: 0.8535, Test Loss: 0.8418
Epoch 22/30, Train Loss: 0.8530, Test Loss: 0.8414
Epoch 23/30, Train Loss: 0.8530, Test Loss: 0.8408
Epoch 24/30, Train Loss: 0.8526, Test Loss: 0.8410
Epoch 25/30, Train Loss: 0.8528, Test Loss: 0.8419
Epoch 26/30, Train Loss: 0.8517, Test Loss: 0.8406
Epoch 27/30, Train Loss: 0.8512, Test Loss: 0.8401
Epoch 28/30, Train Loss: 0.8512, Test Loss: 0.8406
Epoch 29/30, Train Loss: 0.8512, Test Loss: 0.8393
Epoch 30/30, Train Loss: 0.8511, Test Loss: 0.8392
```

Visualization:



4.2 Saving Full model weights:

The full model file (full_model.pth) saves both:

- Model Architecture (Neural Network structure)
- Model Weights (Learned parameters after training)

---

5. Recommendation System

The trained model is used to recommend the top 5 pratilipi IDs for a user based on engagement score.

5.1 Personalized Recommendation

For a given user ID, the model predicts engagement scores for all available pratilipis and selects the top 5 recommendations.

Output:

- **Actual data and Top 5 recommendations of user ID 21:**

	user_id	pratilipi_id
185564	21	100259
405700	21	132667
922559	21	134270
2060743	21	82329
2163151	21	135019
2593881	21	133995
2612881	21	131391
3041211	21	72463
3123340	21	134076
3161778	21	133062

```
Top 5 unique recommendations for user ID 21:
         pratilipi_id  predicted_score category_name
185564         100259      1605.691284        social
2060743         82329      1598.160278        social
3041211         72463      1586.792480        social
3123340        134076      1524.476074        horror
405700         132667      1522.555542        horror
```

- **Actual data and Top 5 recommendations of user ID 71219:**

	user_id	pratilipi_id
277989	71219	15783
286929	71219	15753
629851	71219	77968
913636	71219	61327
3634350	71219	114088
3704648	71219	103731
3941522	71219	4713
3974670	71219	4399

```
Top 5 unique recommendations for user ID 71219:
         pratilipi_id  predicted_score category_name
3704648        103731      1646.298340       romance
277989          15783      1591.840698         women
913636          61327      1586.878174        social
3941522          4713      1553.445435         women
3974670          4399      1400.973877        social
```

- **Actual data and Top 5 recommendations of user ID 29:**

	user_id	pratilipi_id
582020	29	133022
1240325	29	120810
1330847	29	64515
1706868	29	64803
2419085	29	65522
2658199	29	104333
3007436	29	111904
3129532	29	117078
3211435	29	125125
3315736	29	117857

```
Top 5 unique recommendations for user ID 29:
         pratilipi_id  predicted_score category_name
2658199        104333      1736.544922        horror
582020         133022      1678.809937       romance
3211435        125125      1641.837891        horror
3315736        117857      1637.205444        horror
3129532        117078      1636.887573        horror
```

5.2 First-Time User Recommendations

For new users, recommendations are generated based on selected categories.

Input: Social and Romance

Output:

🔹 Recommendation for 'social':

	pratilipi_id	categories
3851401	719	social
3094213	385	social
2722432	688	social
3580723	1014	social
3303350	851	social

🔹 Recommendation for 'romance':

	pratilipi_id	categories
3901564	1125	romance
3580722	1014	romance
3158012	2360	romance
1173799	3696	romance
1231827	3593	romance

---

6. Model Evaluation

After training, the model is evaluated using three standard regression metrics:

6.1 Performance Metrics

```
🔹 RMSE: 0.9161
🔹 MAE: 0.6104
🔹 R² Score: 0.0884
{'RMSE': 0.91606504, 'MAE': 0.6103892, 'R2_Score': 0.08838980543760289}
```

6.2 Interpretation

- **RMSE:** Measures error magnitude.
- **MAE:** Shows the average error per prediction.
- **R^2 Score:** Indicates variance explained by the model.

---

7. Conclusion & Future Work

Key Takeaways

- **The model effectively predicts user engagement (effective_read_words).**
- **Personalized recommendations are based on user interactions.**
- **First-time users receive category-based recommendations.**
- **The model performance is moderate and can be improved.**

Future Enhancements

- **Improve feature selection to increase predictive power.**

- **Experiment with different architectures (e.g., deeper networks or transformer-based models).**
- **Fine-tune hyperparameters to optimize generalization.**
- **Implement user feedback loops for better personalization.**