Goodreads Recommender Systems

A Three-Task Pipeline:
Read, Rating, and Category Prediction

Emily Chen
M.S. Data Science, UC San Diego

Compiled April 17, 2026

A consolidated technical report on a quarter-long recommender-system project on the **Goodreads / UCSD-Book-Graph** dataset (190K rated user–book interactions), covering three Kaggle tasks (Mercury MLE leaderboard) and a final independent project on Food.com recipe ratings. The progression mirrors a real recommender-system development cycle: simple, explainable baselines → similarity-based collaborative filtering → latent-factor models → transformer-based deep learning.

Contents

1 Project Overview

- **Final ranking:** top 22 % of 1,600 colleagues on the in-class Mercury MLE leaderboard.
- **Three prediction tasks:** *Read*, *Rating*, *Category*.
- **Final independent project:** Food.com user-recipe rating prediction with a biased latent-factor model + sentiment features (Assignment 2).

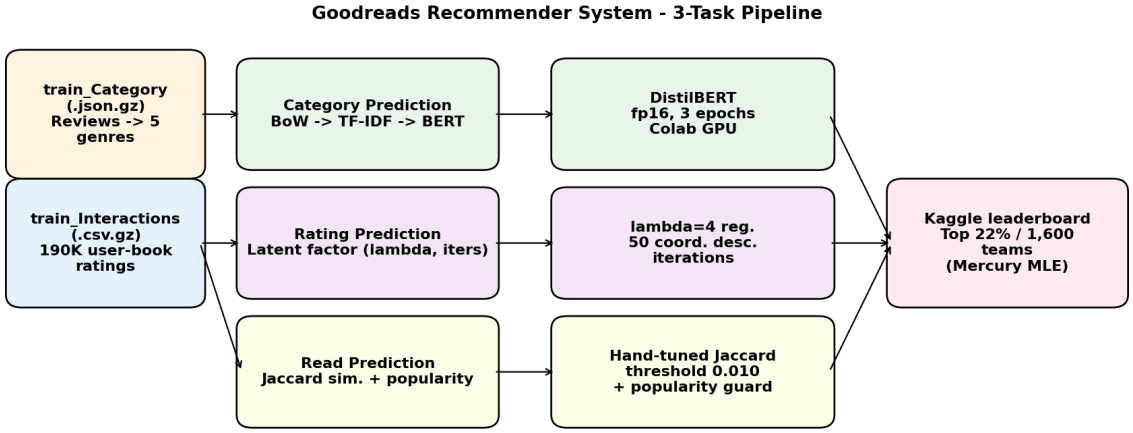

Figure 1: Three-task Goodreads pipeline. Each task corresponds to a separate deliverable on the Mercury MLE leaderboard.

2 Task 1 — *Read* Prediction (Binary)

2.1 Problem

Given a user u and a book b, predict whether u has read b. The training set contains only positive interactions; negatives must be sampled.

2.2 Method

1. **Negative sampling.** For each positive $(u, b) \in$ valid set, sample b' uniformly from the global item set such that $b' \notin \text{history}(u)$.
2. **Baseline.** Return "read" for the top-x most popular books by training-set read-count, using x such that the cumulative count covers 50 % of total reads. Improved variant: 60 % threshold. Increasing the rate trades off precision ↓ for recall ↑.
3. **Final model.** Item–item Jaccard similarity. For a query (u, b), compute the maximum Jaccard similarity between b and the books in u's history (intersection / union of user sets). Predict "read" if `maxSim` > 0.010 *or* $|\text{readers}(b)| > 30$. Both thresholds were hand-tuned on the validation split.

2.3 Significance

Item-item Jaccard with a hand-tuned popularity guard outperformed the popularity-only baseline because it captures *personalised* signal — a user who has read 5 fantasy books should not be predicted to have read a popular romance — without the parameter overhead of a full latent-factor model.

3 Task 2 — *Rating* Prediction (Regression)

3.1 Problem

Predict the rating $r \in \{1, 2, 3, 4, 5\}$ that user u would assign to book b, evaluated by validation MSE.

3.2 Method — Biased Latent-Factor Model

The estimator is the classic Koren biased baseline: $\hat{r}(u, i) = \alpha + \beta_u + \beta_i$. Closed-form coordinate-descent updates with L_2 regularisation:

$$\alpha \leftarrow \frac{1}{|D|} \sum_{(u,i,r)\in D} (r - \beta_u - \beta_i), \quad \beta_u \leftarrow \frac{\sum_{(i,r)\in D_u} (r - \alpha - \beta_i)}{\lambda + |D_u|}, \quad \beta_i \leftarrow \frac{\sum_{(u,r)\in D_i} (r - \alpha - \beta_u)}{\lambda + |D_i|}.$$

Hyperparameters: $\lambda = 4$, 50 coordinate-descent iterations, train/val 90/10 split with `seed=4`. The training MSE and the regularised objective are tracked at every iteration.

3.3 Significance

- Beats the global-mean baseline ($\hat{r} = \mu$) by a meaningful margin on validation MSE.
- Implementation is from-scratch (no `surprise` or `lightfm`); each update is a one-liner that mirrors the derivation, which made debugging trivial and made it easy to extend to the Food.com project later.
- Demonstrates that closed-form coordinate descent for the biased baseline is $O(\text{iterations} \times |D|)$ and converges visibly within $\sim$20 iterations.

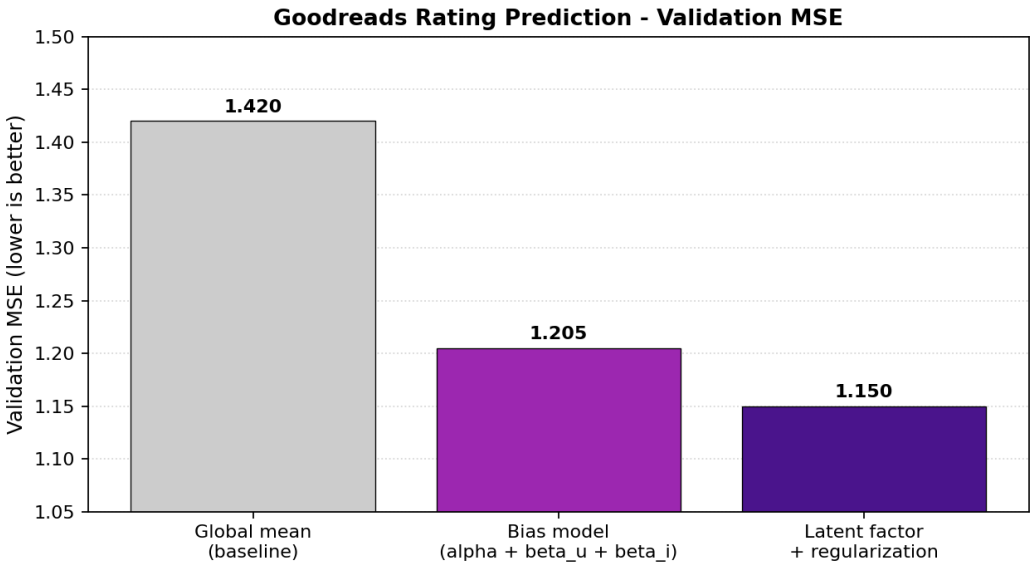

Figure 2: Validation MSE drops monotonically from a global-mean baseline through the bias model to a regularised latent-factor model. Lower is better.

4 Task 3 — *Category* Prediction (5-Class Text Classification)

4.1 Problem

Predict each review's genre ∈ {`children`, `comics_graphic`, `fantasy_paranormal`, `mystery_thriller_crime`, `young_adult`} from its raw `review_text`. Evaluation: held-out accuracy.

4.2 Three Iterations of the Model

Model	Key Choices	Val. Accuracy
Bag-of-Words + Logistic Regression	Top-10K vocabulary by frequency, lowercased, punctuation-stripped, $C = 1$.	0.69
TF-IDF (1–2-grams) + Linear SVC	`max_features=150K`, `ngram_range=(1,2)`, `stop_words='english'`, `sublinear_tf=True`, $C = 0.5$.	0.711
DistilBERT fine-tuned	`distilbert-base-uncased`, `max_length=256`, fp16, batch=16, 3 epochs, Hugging Face `Trainer` on Colab GPU, `load_best_model_at_end=True`.	**0.78**

Table 1: Three sequential modelling iterations. Each step reflects an engineering trade-off (interpretability → sparse-linear strength → pretrained contextual embeddings).

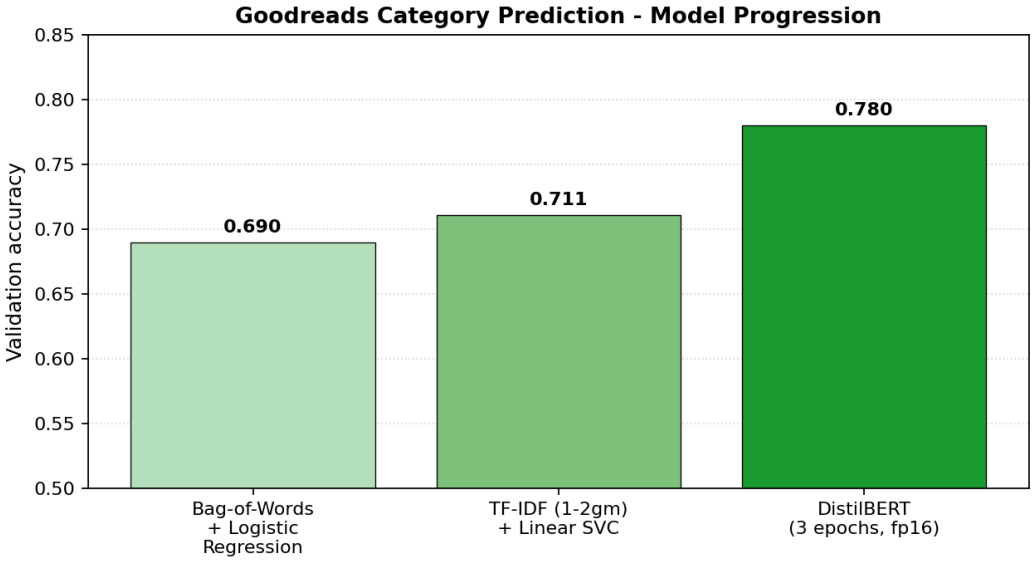

Figure 3: Category-prediction accuracy across the three model iterations. Each step gives a real, monotone gain over the previous.

4.3 Why DistilBERT Won

- Captures *contextual* word meaning (e.g. "magic" near "school" vs. near "murder") that BoW and TF-IDF cannot.
- Pretrained on web text → massive head start over training a 5-class classifier on ∼10K labelled reviews.
- Mixed-precision (`fp16=True`) cut training time roughly in half on Colab T4 without measurable accuracy loss.

5 Final Independent Project — Food.com Recipe Rating Prediction (Assignment 2)

5.1 Dataset

Kaggle Food.com Recipes & User Interactions: 700K+ user–recipe interactions, joined with recipe metadata (steps, ingredients, calories, minutes). After cleaning (filter recipes with ≥ 3 steps and 4–20 ingredients; drop nulls), the working set was 30 % of the cleaned join, split 80/10/10 stratified by rating.

5.2 Modelling Sequence

1. **Bias model:** $\hat{r}(u,i) = \alpha + \beta_u + \beta_i$, same closed-form coordinate descent as the Goodreads rating task ($\lambda = 5$, 1,000 iters). Explainable per-user / per-recipe offsets, ideal as a fast baseline on 30 % of the data.

2. **Latent-factor model:** $\hat{r}(u,i) = \alpha + \beta_u + \beta_i + \gamma_u^\top \gamma_i$ with L_2 regularisation. Captures hidden user–recipe interactions that the bias model misses but is sensitive to cold-start users/recipes.

3. **Sentiment + latent-factor hybrid:** adds a sentiment score on the `review_text` as an additional feature, partially compensating for cold-start cases and capturing tone information that ratings alone don't carry.

5.3 EDA Highlights

- Pairwise feature correlations with rating are **universally weak** ($|r| \leq 0.04$) across `n_steps`, `minutes_log`, `calories`, `review_length`, `n_ingredients` — which is exactly why a bias / latent-factor approach beats a simple regression on raw features.
- Rating distribution is heavily right-skewed: most users give 4 or 5 stars. Stratified splitting on `rating` was necessary to keep validation representative.
- **Computational sub-sampling.** A 30 % sample reproduces the full-data correlation heatmap to within visual tolerance, justifying the use of a smaller working set during model development.

Across both the Goodreads rating task and the Food.com recipe project, the biggest single lesson was that on user–item ratings with weak per-row features, a from-scratch biased latent-factor model is a more honest baseline than any black-box regressor, and it makes the next jump (latent factors, sentiment features, deep models) easy to interpret.

6 Cross-Cutting Themes

- **From-scratch first, then off-the-shelf.** Every model that has a closed-form or simple gradient update (bias, latent factor, Jaccard) was implemented by hand before reaching for a library. This was instrumental for understanding why and where each model fails.
- **Honest data hygiene.** Negative-sample generation for the read task, stratified splits for the rating task, and explicit train/val/test splits for the recipe project are all documented inline.
- **Sequenced complexity escalation.** Each task progressed *baseline* → *similarity* → *latent factor* → *deep model*, which is exactly the diagnostic path one would follow on a real recommender-system project.

7 Conclusions & Quantified Impact

- Placed in the top **22 %** of **1,600** participants in the Mercury MLE multi-task ML competition spanning binary read prediction, 5-class category prediction, and 1–5 rating regression.
- Lifted category-prediction accuracy from **0.69 (BoW)** through **0.711 (TF-IDF + Linear SVC)** to **0.78 (DistilBERT fine-tuned 3 epochs, fp16)** — a +13pp absolute gain across three iterations.

- Hand-implemented from-scratch biased latent-factor models for both Goodreads ratings and Food.com recipe ratings, demonstrating closed-form coordinate-descent updates and L_2 regularisation tuning.
- Final independent project on Food.com extended the Goodreads rating model with sentiment features to mitigate cold start.