Recommendation System Optimization and Performance Analysis

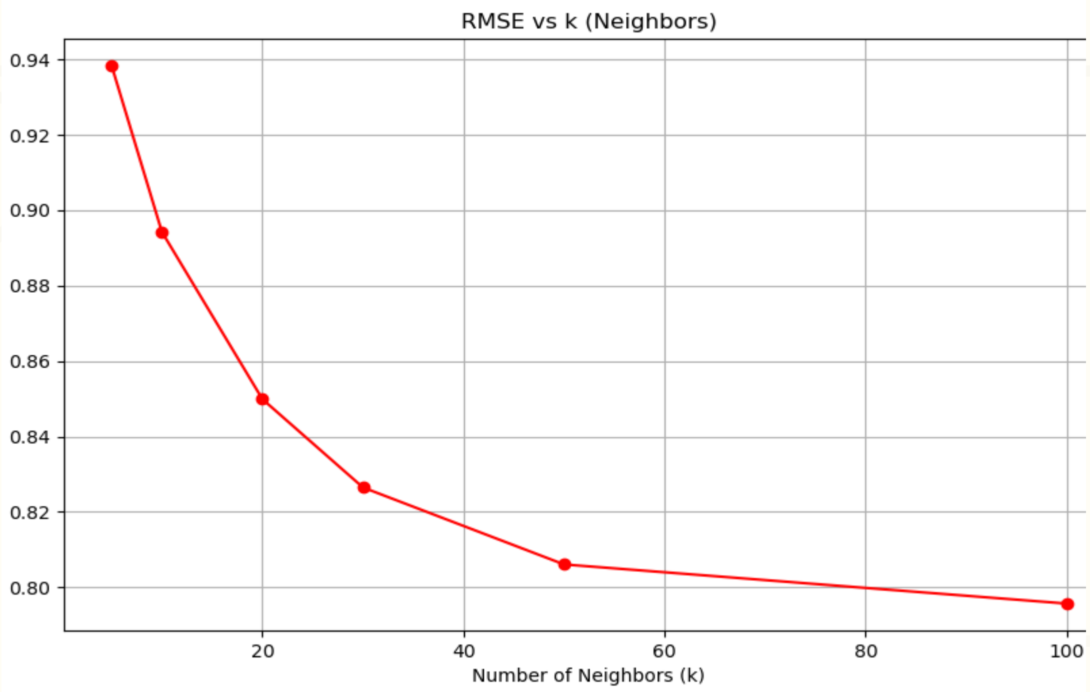
MovieLens 100K Dataset

Project Summary

- Implemented 5 methods for recommendation with MovieLens 100K dataset.
- Performance each method is improved by tuning the parameters in different similarities.
- The last method is hybrid approach (Method 5) achieved the best results with RMSE of 0.7173.

General optimization strategy

- Parameter isolation.
- Cross validation.
- Error metric: RMSE than MAE.
- Visual analysis.



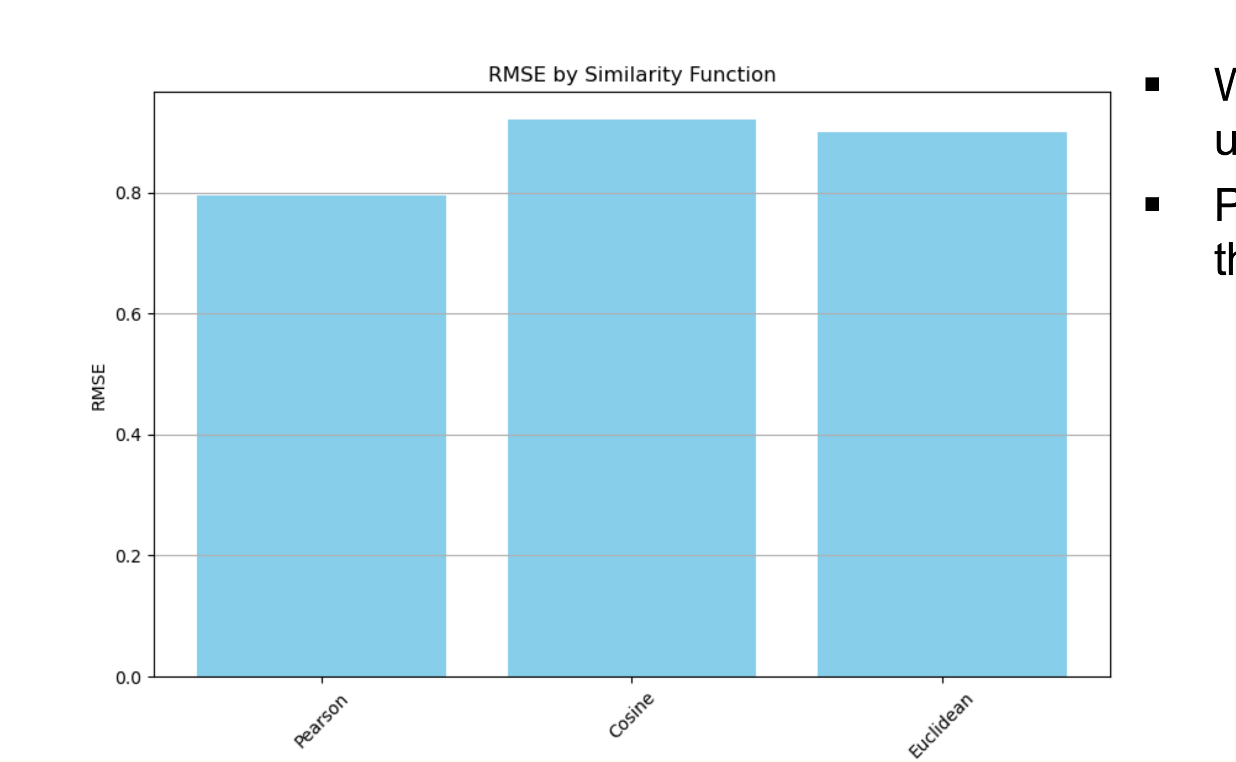
Method 3: User-based collaborative filtering - Neighborhood size (k)

- We set default similarity and weight, then tune K with different values.

As k increases, it improves performance while getting small RMSE, optimal at k=100 (RMSE: 0.795).

- More neighbors help reduce the rating noise.

Method 3: User-based collaborative filtering - Similarity function



- We test 3 similarities using size k = 100.
- Pearson achieved the lowest RMSE.

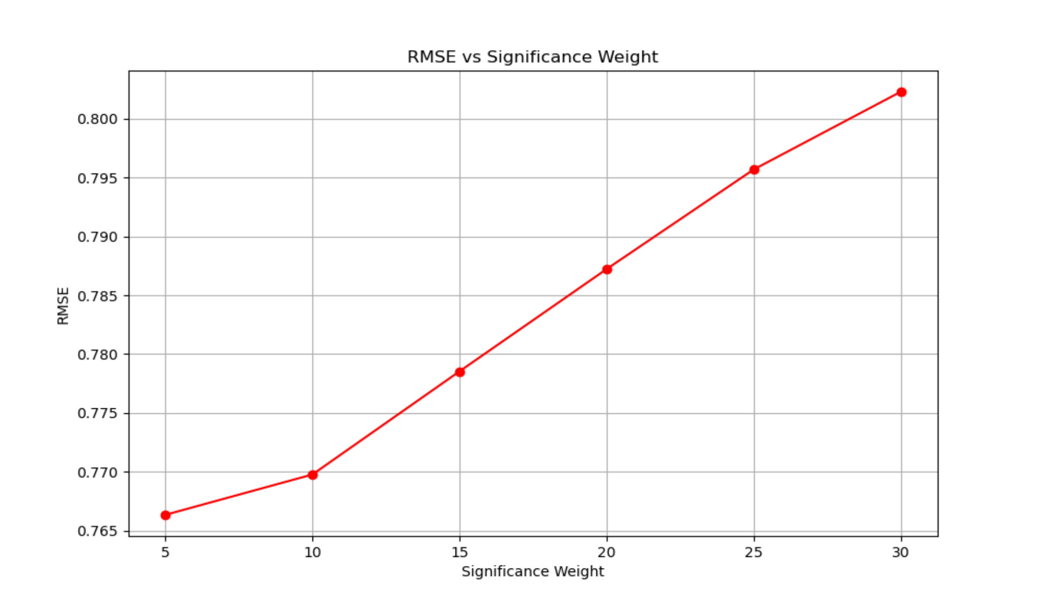
Method 3: User-based collaborative filtering - Similarity Function

Why pearson is better for users:

- Rating normalization: pearson subtracts user’s mean rating before comparing them which is useful for users who consistenly rate higher or lower than others.
- Captures the pattern of preference: pearson reflects relative preferences. For instance, two users might both prefer action movies over comedies even their rating styles are different.
- More meaningful similarity: based on comparing center ratings, score is more accurate between users.

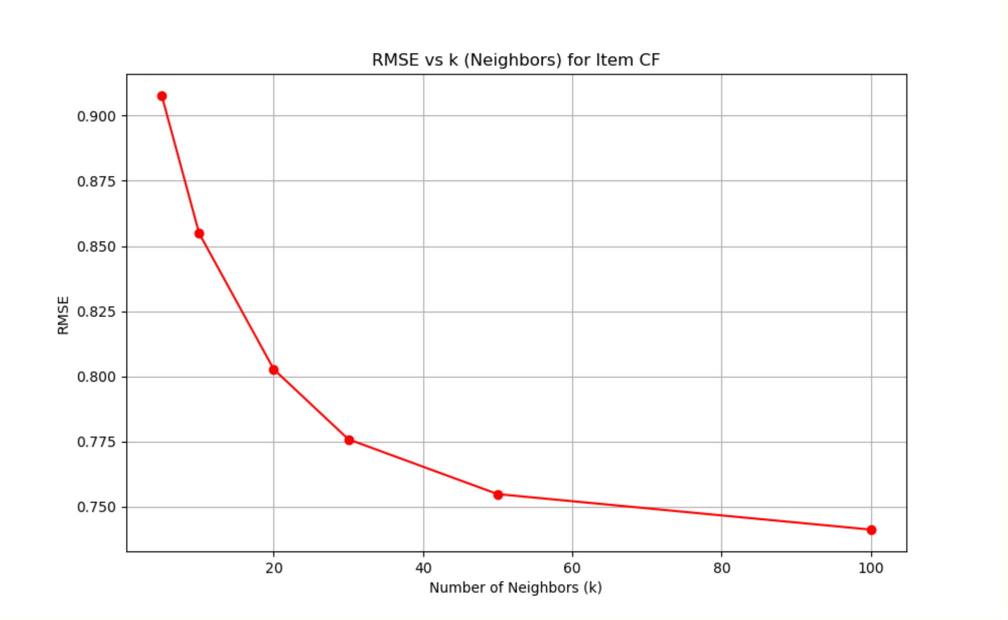
Method 3: User-Based collaborative filtering - Significance Weight

- Continue tuning weight with k and similarity that we have found.
- Lower weights consistently archived better performance.
- Optimal value for weight at 5 (RMSE: 0.766).
- With few co-rated items between users provides valuable similarity information.
- When the weight gets larger, it reduces ability to find meaningful patterns.

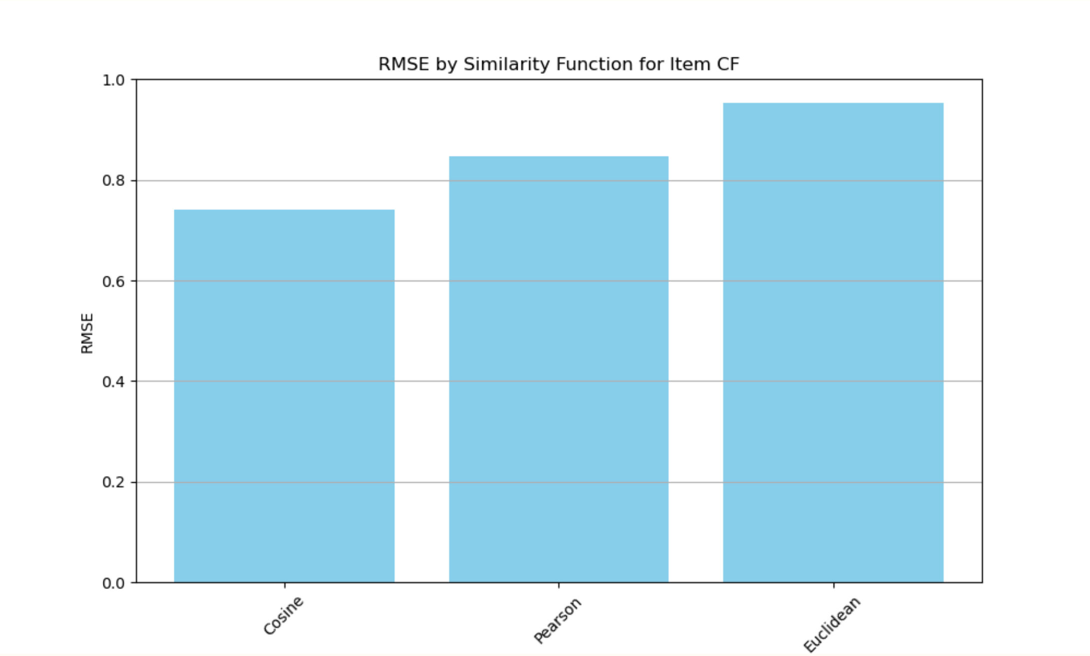


Method 4: Item-Based CF - Neighborhood Size (k)

- Using default cosine similarity and weight to tune k.
- Performance improves with larger neighborhood size, optimal at k=100 (RMSE: 0.740).
- With a larger neighborhood helps the model understand item relationships more, improving prediction accuracy.



Method 4: Item-Based CF - Similarity Function



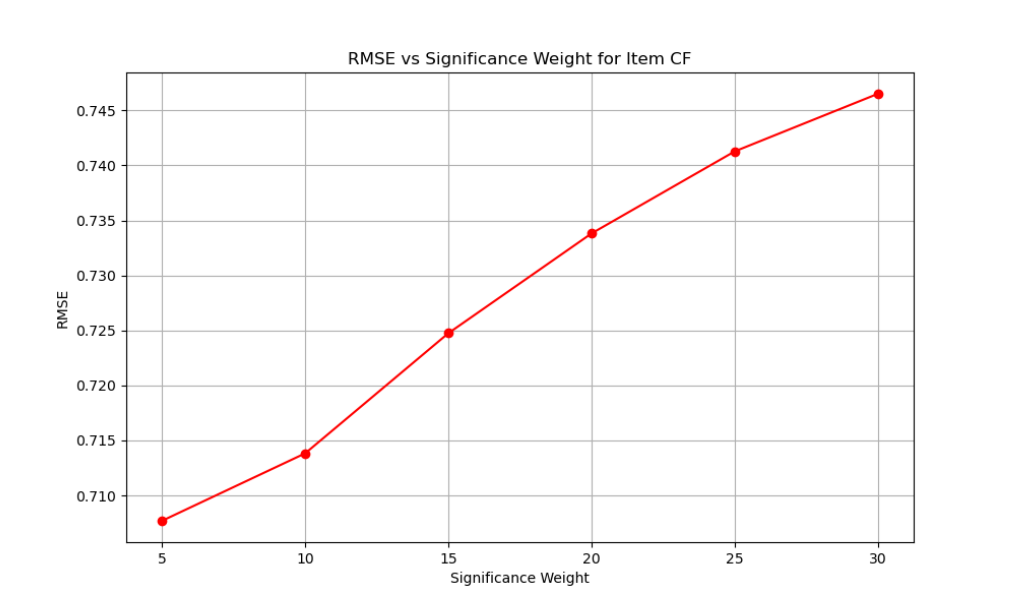
- Test 3 similarities with k=100.
- Cosine similarity performed best performance.

Method 4: Item-Based CF - Similarity Function

Why cosine works better for items:

- Focusing on rating directions: the users tend to rate items consistenly which make raw values valuable, and cosine compares angle between rating vectors without substracting the mean.
- Capturing the popularity better: items are rated on a similar scale from most users.
- Consistent rating trends: item ratings tend to be rated stably across the users, cosine effectively identifies similarity based on sharing popularity.

Method 4: Item-Based CF - Significance Weight



- Lower weights produce better results.
- Optimal value at 5 (RMSE: 0.708).
- Valuable similarity information still be contained in few co-rated items.

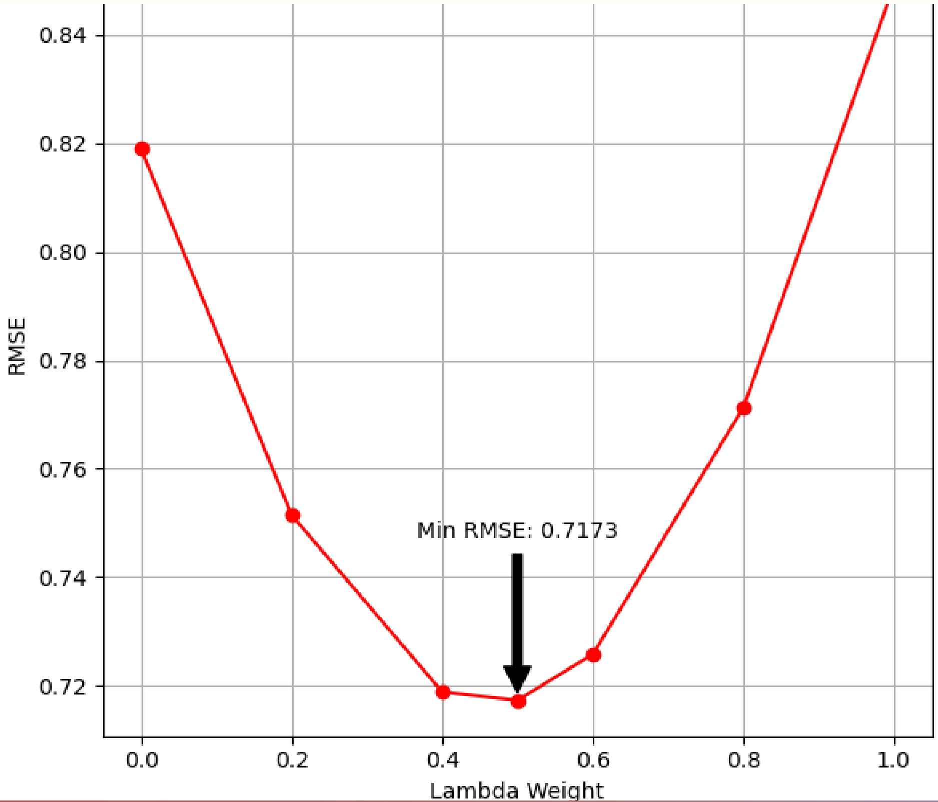
Method 5: Hybrid Approach Optimization

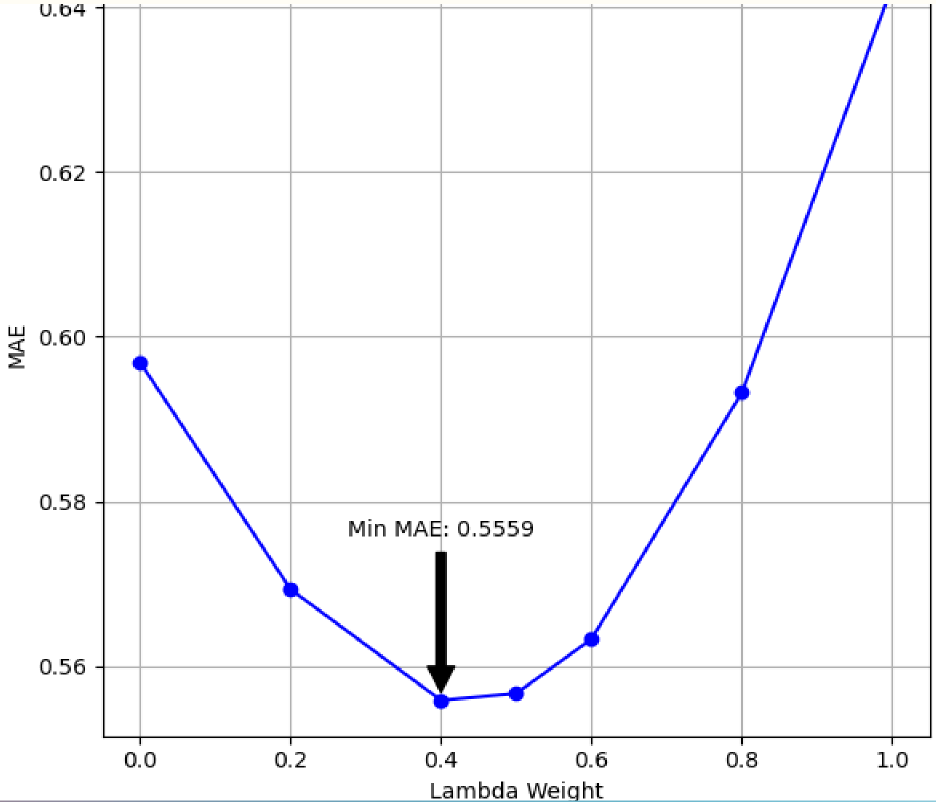
Optimization strategy:

- Test lambda from 0 to 1.
- Combine prediction form user-based and item-based at previous methods.
- Best performance at lambda equals 0.5 which contributes from both models

Performance:

RMSE = 0.7173, best all methods.





Method 5: Hybrid Approach Optimization

Why it works:

- ✓ User-based captures individual preferences.
- ✓ Item-based provides stably estimates across similar items.
- ✓ Hybrid method leverages both to mitigate the individual weaknesses.

The role of optimization in enhancing model performance

Understanding individual parameter impact:

- Isolating each parameter which reveals the specific impact on perfomance.

Cross-validation for reliable performance estimates

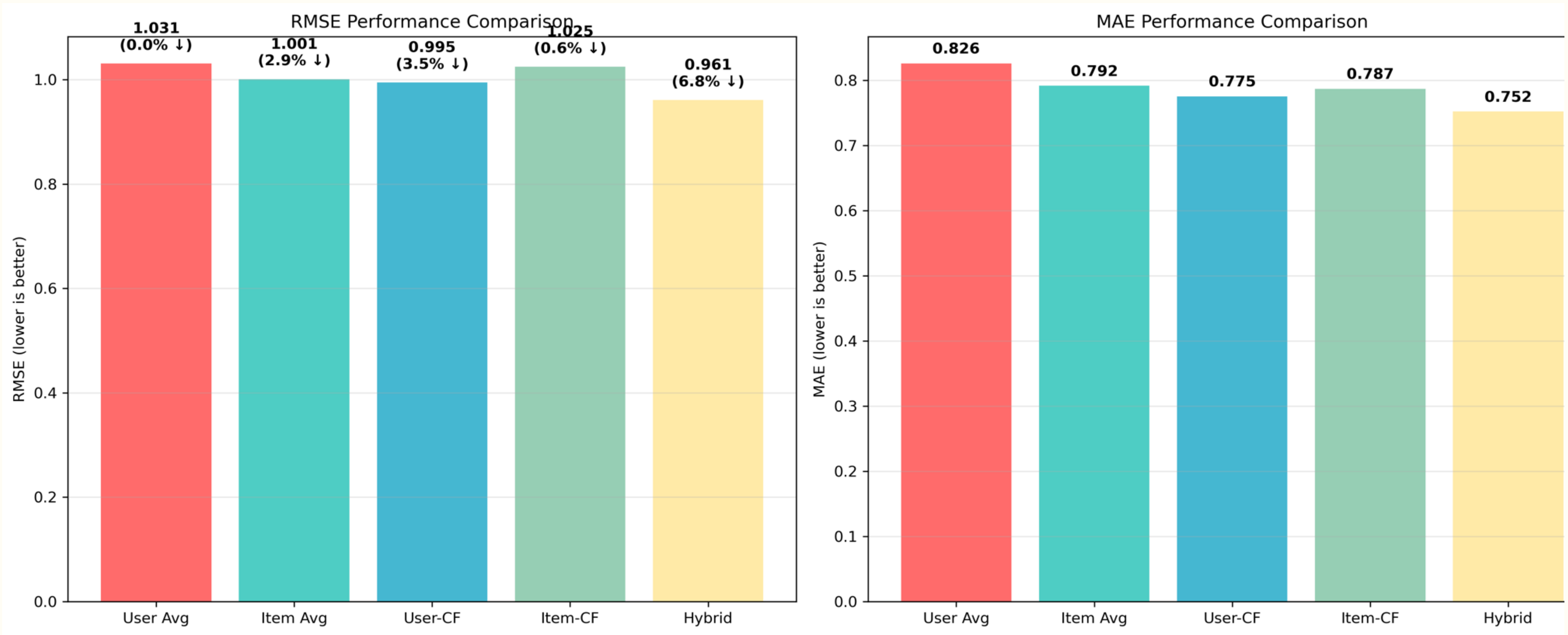
- Providing reliable performace on multiple datasets.
- Reveal how models generalize to unseen data, which is big parts of recommendation.
- Discorvering optimal parameters might be missed with single dataset.

Why RMSE is a better fit than MAE for recommenders

- It squares errors before averaging which penalizes large predictrion mistakes more heavily.
- It helps avoid significant prediction errors which is critical to mainting users trust in recommendation.
- It is reliable in industries.

Identifying optimal values through visualization

- It is more ituitive which is accessible evidence.
- Visual performs the trends underlying the data.



Comparative Analysis of All 5 Methods

From the best to worst performance (by RMSE):

1. Method 5: Hybrid 6.8% improvement over baseline.
2. Method 3: User-based with 3.5% improvement.
3. Method 2: Item average with 2.9% improvement.
4. Method 4: Item-based with 0.6% improvement.
5. Method 1: User-based as baseline.

Method 1: User average

- Approach: Predicts ratings based definitely on the user's average rating behavior.
- Performace: the result is reasonable with RMSE is 1.0311, and MAE is 0.826.
- Strengths:
  - ❑ Simple to implement.
  - ❑ Handle new items with no ratings yet.
- Weaknesses:
  - ❑ Ignore completely information on items.
  - ❑ Can not capture preferences on items from users.

Method 2: Item average

- Approach: predicts rating on each item's average across all users.
- Performace: better than user average with RMSE 1.0013, and MAE is 0.79
- Strengths:
  - Simple implementation.
  - Leverage the popularity and quality for items.
  - Can handle for new users with no rated items yet.
- Weaknesses:
  - Ignores references on specific users.
  - Can not capture the personalized recommendations.

Method 3: User-based collaborative filtering

- Approach: Predicts ratingby finding the similarity between users, then using their rates.
- Performace: improve a bit over baseline (method 1) with RMSE 0.99, and MAE 0.77.
- Strengths:
  - ✓ Show the personalized recommendations.
  - ✓ Capture the user preference patterns effectively.
  - ✓ Could recommend item which is less popular but high relevent to that user.
- Weaknesses:
  - ✓ Computional complexility for finding similary.
  - ✓ Sensitive to parmater tuning.

Method 4: Item based collaborative filtering

- Approach: predicts ratings using similarity between items.
- Performace: generaly well but slightly worse than the user-based RMSE 1.02 and MAE: 0.78.
- Strengths:
  - Stability than user-based with the new ratings due to the valid longer between items.
  - Lower complexity in predictions because of we focus on the similarities between items and only look the items that user has already rated.
- Weeknesses:
  - Depends on ratings between items.
  - Prefer to recommend with popular items.
  - Less effectively recommend relevant items for that users with few ratings.

Method 5: Hybrid approach

- Approach: Optimized lamda to weight balance between user-based and item-based.
- Performance: best perfomance in all methods, improve 6.8% over the baseline significantly.
- Strengths:
  - Strength both collarorative filtering by combination.
  - Advantage for data sparsity when few items are rated.
  - Better for handling the edge cases when ensuring more consistent predictions.
- Weaknesses:
  - Complexity for implementation.
  - More parameters are tuned to get the optimal perfomance.
  - More complex to implement.

Why certain methods perform better than others?

Why hybird method achieved the best performance (6.8% improvement over base line)?

- Combine information: it strengths two different methods which covers individual weakness of both approaches.
- Mitigating the errors: when one method has larger errors in predictions, another method could mitigate or reduce extremely errors.
- Tuning indetified lambda 0.5 indicates balance between valuable information between two methods.

Why certain methods perform better than others?

Why user-based collaborative filtering (method 3) performs better than item-based collaborative filtering (method 4)?

- Normalization in ratings: some users might consitently rate higher or lower than others makes the similarity meaningful.
- Coverage: the user–based might achive better coverage because of finding the similarity between users migt be easier than the items.

Why certain methods perform better than others?

Why the item average (method 2) outperfoms the more complex item-based collaborative filtering (method 4)?

- Noise sensitivity: the more complex item-based might me sensitive to noise.
- Consistency on ratings: items may have relatively consistent ratings across different users which makes the item average a strong prediction.
- Complexity similarity: additional complexity provides less significantly advantage than simple item averages.

Why certain methods perform better than others?

Why the user average is the weakest?

- No personalization: ignore completely the information on specific items, treating all items equally.
- No collaborative information: it does not spot the potential patterns how similar users react to similar items.
- Ignore items quality: unlike item average, it does not count on the popular items.
- Rating scale: it captures more of user's rating tendencies without recognizing that different users have different rating scales.

- Thank you for listening!