Movie Recommendation Service Review

Machine Learning in Production - Team 25

Adnaan Yunus
Ajay Mittur
Glenn Xu
Jiaji Li
Jiale Jiang
Joey Xie

Team Information

- Skills prior to project development
  - Adnaan Yunus - DevOps, Full Stack Web Development
  - Ajay Mittur - Machine Learning, Software Engineering
  - Glenn Xu - Machine Learning
  - Joey Xie - Machine Learning
  - Jiaji Li - DevOps, Architecture design
  - Jiale Jiang - Full Stack Web Development
- Team dynamics:
  - Contribution was largely oriented towards what the skill sets were of each member.
  - Frequent code reviews helped team understand different parts of the codebase

Goals and Achievements

- Created a model that achieved 90 percent plus accuracy in offline and online evaluation.
- Model memory utilization was low and inference times were considerably fast
- Docker orchestrated servers of our service achieved a 95-99 percent uptime.

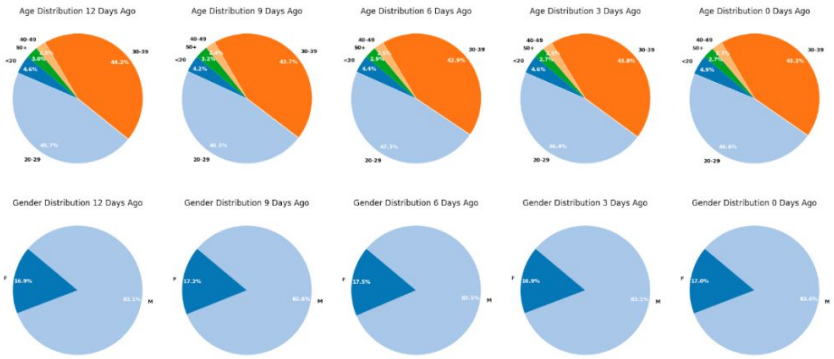
Dealing with data – Data Preprocessing and Storage

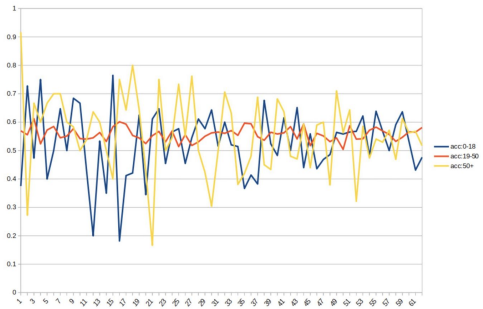
- Data Source: API (user/movie); Kafka Stream
  - Build a simple model w/ less attributes

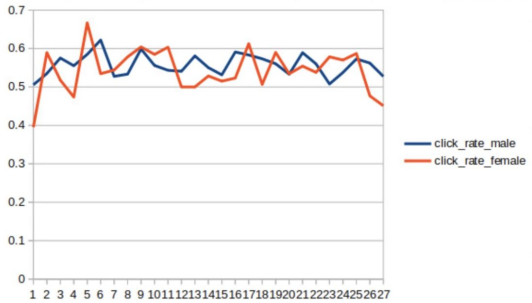
- Storage: CSV file, long response time
  - → Choose one Database instead? E.g. Postgres SQL? TinyDB?

Monitor Data Drift and Ensure Fairness

- No obvious data drift detected
- Fairness
  - Fewer female users
  - Fewer young/old users
  - Reasons?
    - Recommendations result?
  - Fairness
    - Not stable recommendation?
    - Not enjoying the same quality service







Model Comparison

Model vs. Metric	**SVD**	**XGBoost**
Prediction Accuracy (RMSE/MAE)	0.90/0.71	0.91/0.72
Training Cost	6.32s	24.36s
Inference Cost	0.86s	0.22s
Disk Size/Memory Size	115MB/94MB	0.64MB/0.0034MB

SVD Pros:

- Low training cost
- No need for feature engineering
- Learn embeddings, can be used to find similar users or movies.

XGBoost Pros:

- Low Inference cost
- Low memory size
- Use of user demographics, movie metadata information

Online Testing

Accuracy Metric: Click Rate(CTR)

$$\text{Click Rate} = \frac{\text{Clicks}}{\text{Impressions}} \times 100$$

Accuracy Metric: Mean Reciprocal Rank (MRR)

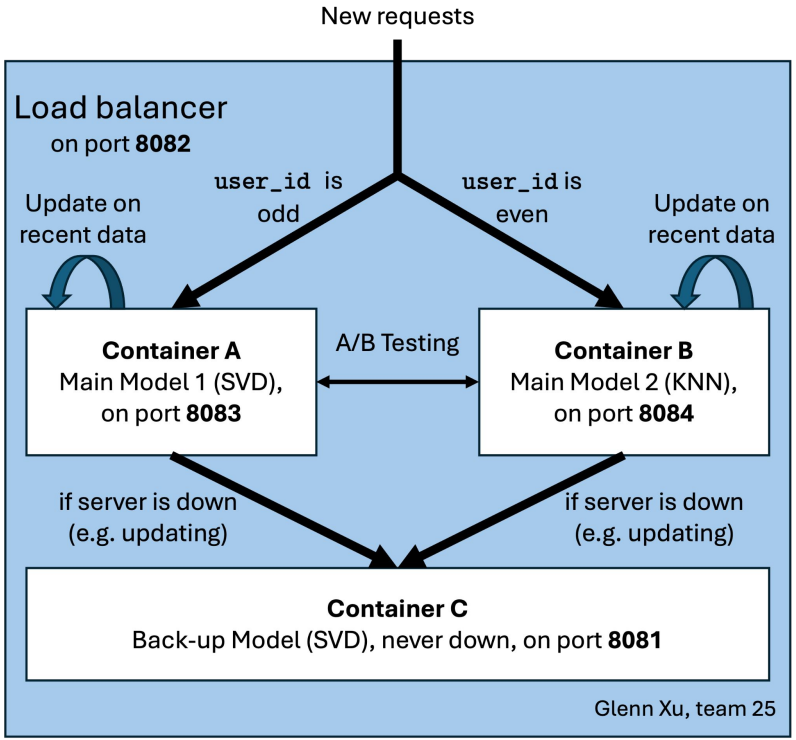
$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i}.$$

```
Evaluation round 61:
Time:  2024-03-18T14:39:44      Click Rate:  0.6567164173659067         Mean Reciprocal Rank:  0.36630894484128873

Evaluation round 62:
Time:  2024-03-18T14:53:02      Click Rate:  0.6403654479731183         Mean Reciprocal Rank:  0.34755167630163347

Evaluation round 63:
Time:  2024-03-18T15:06:03      Click Rate:  0.6613756607924376         Mean Reciprocal Rank:  0.3808166725735258

Evaluation round 64:
Time:  2024-03-18T15:18:36      Click Rate:  0.6599450039780522         Mean Reciprocal Rank:  0.37287417838712594

Evaluation round 65:
Time:  2024-03-18T15:30:49      Click Rate:  0.6595121944785246         Mean Reciprocal Rank:  0.35707114295705555

Evaluation round 66:
Time:  2024-03-18T15:42:44      Click Rate:  0.6462450586499555         Mean Reciprocal Rank:  0.3536632875238851

Evaluation round 67:
Time:  2024-03-18T15:54:05      Click Rate:  0.6059063130286086         Mean Reciprocal Rank:  0.3392944298107449

Evaluation round 68:
Time:  2024-03-18T16:05:33      Click Rate:  0.6351626009805258         Mean Reciprocal Rank:  0.3502947204939288

Evaluation round 69:
Time:  2024-03-18T16:16:39      Click Rate:  0.6417582410530129         Mean Reciprocal Rank:  0.33900195808286987

Evaluation round 70:
Time:  2024-03-18T16:27:55      Click Rate:  0.6549586770093402         Mean Reciprocal Rank:  0.3589234448014326

Evaluation round 71:
Time:  2024-03-18T16:38:38      Click Rate:  0.6521264987435916         Mean Reciprocal Rank:  0.3525969325982618

Evaluation round 72:
Time:  2024-03-18T16:49:06      Click Rate:  0.611683848096582  Mean Reciprocal Rank:  0.33973181487395004
```

Monitoring

- Use Prometheus and Grafana to monitor online testing click rate, MRR and service availability for both 2 models

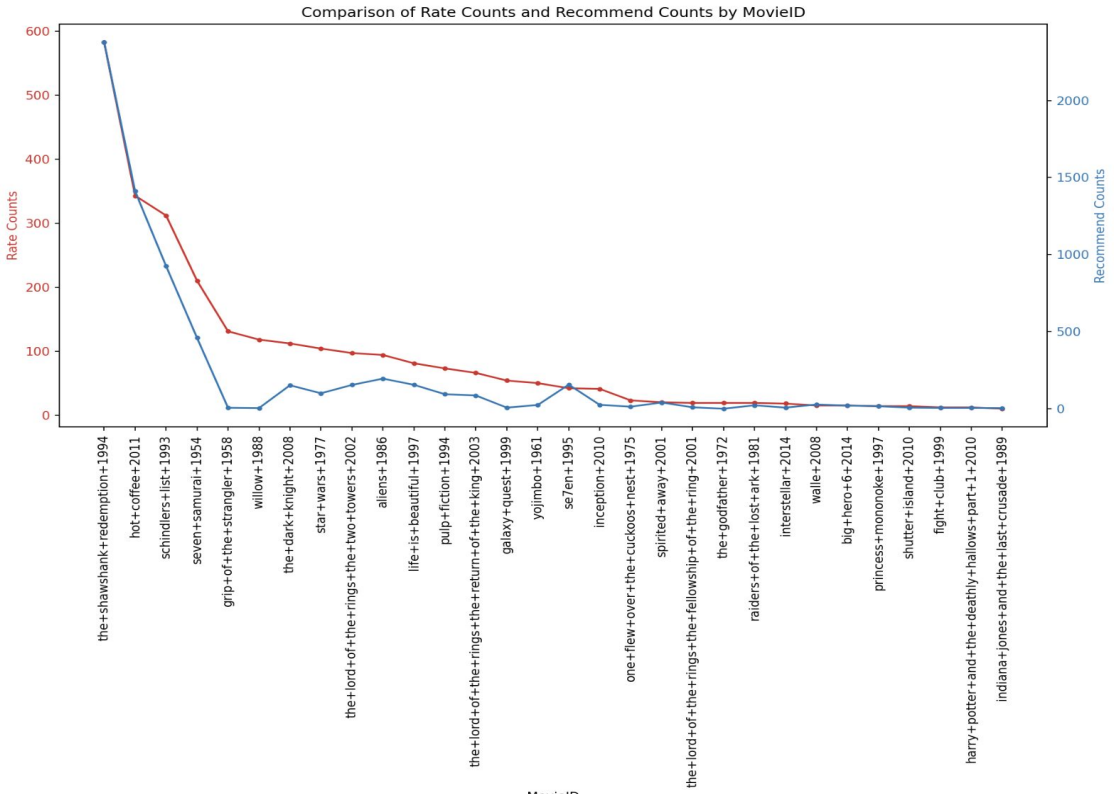
A/B Testing


New requests
Load balancer
on port 8082
user_id is odd
user_id is even
Update on recent data
Update on recent data
Container A
Main Model 1 (SVD), on port 8083
A/B Testing
Container B
Main Model 2 (KNN), on port 8084
if server is down (e.g. updating)
if server is down (e.g. updating)
Container C
Back-up Model (SVD), never down, on port 8081
Glenn Xu, team 25

Feedback loop

- High Concentration of Recommendations: Our analysis showed that a small subset of movies consistently dominated the recommendations, highlighting a significant bias toward popular films.
- Skewed User Ratings: We also found that user ratings were heavily concentrated on the same set of frequently recommended movies, reinforcing the feedback loop that prioritizes popular films and reduces recommendation diversity.

Hybrid approach, New movies, Diversity




```
correlation_coefficient = df_merged['count_rate'].corr(df_merged['count_recommend'])
print("Correlation Coefficient:", correlation_coefficient)
```

[49] ✓ 0.0s

```
Correlation Coefficient: 0.9453035095997545
```

Difficulties - System

- Faced frustration in integrating service like airflow and sql in Docker, which hindered the integration of new improvements to the existing service.
- Service had issues in early development stages:
  - Low uptime
  - Slow model inference time
  - Fixed via different model architecture, caching, and error handling

Difficulties - Teamwork

- Challenging to work smoothly without understanding each other’s code
- Tasks were not entirely mutually exclusive from one another and required communication between different group members.
- Team members needed to understand and implement different technological frameworks or toolkits to accomplish their tasks.

What went well - Teamwork

- We have each milestone a different project manager to keep everything on schedule, and conduct meetings and activities
- Team worked well together in integrating different parts of the codebase to accomplish their respective tasks for each milestone
  - Jenkins and Unit Tests
  - Docker and A/B Testing

What we learned

- **Project Overview:**
  - Key aspects: Deployment, scaling, reliability, monitoring, and feedback loops.
- **Teamwork and Infrastructure:**
  - Collaborative efforts utilizing a shared GitHub repository and a virtual machine.(Pull request)
  - Communication via Slack and task management through Github.
  - Diverse backgrounds in the team to leverage strengths in machine learning and software engineering.
- **Data Collection and Processing:**
  - Use of Apache Kafka for real-time data streaming.
  - API access for movie and user data, with attention to rate limits and data preprocessing.
  - Data features include user behaviors, movie ratings, and streaming logs.
- **Model Development and Deployment:**
  - Exploration of various machine learning models, with a focus on collaborative filtering.
  - Deployment of a model inference service using Flask, handling HTTP requests for movie recommendations.
  - Evaluation based on prediction accuracy, training and inference costs, and model size.
- **Monitoring and Evolution:**
  - Real-time monitoring of service performance and user engagement.
  - Adaptive strategies to handle data drift and feedback loops.
  - Continuous evolution of the recommendation system based on user feedback and system analytics.

In Conclusion

We faced a lot of challenges...

- **Data Management Challenges:**
  - Inefficiencies from using CSV files and manual updates.
  - Proposed shift to database use for robust data handling and simplified access.
- **Server Architecture and Code Management:**
  - Initial non-scalable, messy codebase requiring extensive late-stage refactoring.
  - Recommendation: Establish coding standards and clear project structures from the start.
- **Technical Decisions and Scalability:**
  - Current architecture not equipped for high user loads.
  - Need for scalable system design to accommodate growth without makeshift solutions.
- **Monitoring and Continuous Improvements:**
  - CI/CD pipeline fragility due to incomplete testing.
  - Emphasis on developing comprehensive tests and stable deployment practices.

In Conclusion

But, we pulled through with great teamwork overall

- **General Performance:**
  - Strong collaboration and commitment throughout most milestones.
  - No significant friction or conflicts; compromises made as needed for team harmony.
- **Challenges Faced:**
  - Milestone 3 presented challenges due to high individual workloads, leading to uneven task distribution and stress.
  - Issue was resolved in subsequent milestones, reinforcing understanding of responsibilities.
- **Proposed Changes for Future Projects:**
  - Consider random task assignments to encourage skill diversification and prepare for industry variability.
  - Emphasize the importance of stepping out of comfort zones to foster broader skill development.
- **Key Takeaways:**
  - Importance of clear communication and realistic expectations to avoid overpromising and under-delivering.
  - Recognize the necessity of making sacrifices to maintain professional relationships within the team.