Final Project Presentation

Leo Yuan, Guanhua Ji, Zi Zhu, Catherine Chen

CS 145 Spring 2025

Jun 15, 2025

Team Introduction and Project Overview

Team members: 4 graduating seniors (Math/CS majors)

Project Objective:

Maximize discounted revenue using recommendation systems with 3 recommendation paradigms:

- Content-based recommendation
- Sequence-based recommendation
- Graph-based recommendation

Checkpoint 1: Content-based Recommenders

Approach:

- Models: KNN, Random Forest, XGBoost
- Scoring: Expected Value = P(purchase) × Price
- Tuned via grid search + cross-validation

Outcome:

- XGBoost outperformed all models (test discounted revenue: 3,472.33)
- Revenue-based scoring > ranking-based scoring
- Became our final winning model

Table 8: Detailed Hyperparameter Configurations for Content-Based Models

Model	Parameter	Optimal Value
KNN	metric	cosine
	n_neighbors	10
Random Forest	max_depth	10
	max_features	'log2'
	min_samples_leaf	2
	min_samples_split	2
	n_estimators	100
XGBoost	n_estimators	100
	learning_rate	0.1
	max_depth	5
	subsample	0.8
	colsample_bytree	0.8
	early_stopping_rounds	10
	eval_metric	"logloss"

Checkpoint 2: Sequence-based Recommenders

Approach:

- Models: RNN (GRU), Transformer, AutoRegressive n-gram
- Modeled user behavior over time
- Incorporated item embeddings + price projections

Key Tuning:

- RNN: Reduced hidden size (256→64), tuned dropout to prevent overfitting
- Transformer: Optimized attention heads and feed-forward size
- AutoRegressive: Used n=2, price multipliers, and temporal weighting

Outcome:

- RNN had highest test discounted revenue (3,331.25)

The RNN architecture employs the following detailed configuration:

- **Embedding Layer**: 64-dimensional item embeddings with vocabulary size mapped to contiguous indices
- **Price Projection**: 16-dimensional projected price vector concatenated with item embeddings
- **GRU Layers**: Two-layer architecture with hidden size 64, dropout rate 0.1
- **Output Layer**: Linear transformation to full item vocabulary size
- **Training Configuration**: Adam optimizer, learning rate 0.001, gradient clipping, early stopping

The Transformer implementation includes:

- **Multi-head Attention**: 4 attention heads with 128 hidden dimensions
- **Feed-forward Network**: 256 dimensions with dropout rate 0.2
- **Positional Encoding**: Standard sinusoidal positional embeddings
- **Sequence Length**: Maximum length 100 with padding and attention masks
- **Training**: Cross-entropy loss, Adam optimization, gradient clipping

Checkpoint 3: Graph-based Recommenders

Approach:

- Modeled user-item interactions as bipartite graphs
- Methods:
  - EfficientGraph: NMF + cosine similarity
  - LightningGraph: SVD + Random Forest ensembles

Features:

- Incorporated user spending tiers and price-aware scoring
- Lightweight fallbacks for cold-start scenarios

Outcome:

- EfficientGraph: Solid results (3,139.53 test discounted revenue)
- Good balance of performance + interpretability, but still behind content-based models

Table 9: EfficientGraph Technical Configuration

Component	Configuration
Matrix Factorization	NMF with 32 latent components
Regularization	$\alpha = 0.1$
Max Iterations	100
Graph Weighting	0.25
User Preference Tiers	4 levels (premium, high, medium, low)
Similarity Metric	Cosine similarity
Revenue Boost Factor	4.0

Table 10: LightningGraph Technical Configuration

Component	Configuration
SVD Dimensions	64 for initial embedding
Compressed Features	32 dimensions each (user/item)
Random Forest Models	2 models with 50 estimators each
Feature Set	user avg price, activity, item price, popularity, value
Revenue Boost Factor	4.5
Standardization	StandardScaler normalization

Key Experimental Results

Cross Validation Results:

Table 11: 5-Fold Cross-Validation Results for Top Performing Models

Model	Mean CV Score	Std Deviation	Final Test Score
XGBoost	24,387.04 $\pm$ 1,245.32	1,245.32	28,024.11
KNN	24,484.12 $\pm$ 1,156.78	1,156.78	27,118.17
Random Forest	25,251.16 $\pm$ 987.45	987.45	25,519.30
RNN	22,121.17 $\pm$ 1,567.89	1,567.89	28,157.54
EfficientGraph	25,001.89 $\pm$ 1,023.67	1,023.67	25,787.36

Feature Importance Analysis:

Table 12: Top 10 Feature Importance Scores for XGBoost Model

Feature Name	Importance Score
item_price	0.234
user_avg_purchase	0.187
item_popularity	0.156
user_total_interactions	0.143
item_category_electronics	0.098
user_price_preference	0.087
item_rating_avg	0.065
user_activity_recent	0.054
item_discount_history	0.043
user_session_length	0.032

Performance Summary:

Method	Test Discounted Revenue	Test Total Revenue	Test Avg Revenue	Performance Change
ContentBased	**2831.22**	23055.59	4611.12	-1.33%
Graph	2511.50	20663.14	4132.63	-9.69%
AutoRegressive	2371.08	18983.81	3796.76	-13.36%

Key Experimental Results

Table 4: Checkpoint 1 Detailed Performance Results

Method	Train Revenue	Test Revenue	Test Precision@K	Test NDCG@K	Test MRR	Test Disc. Revenue
GB	24,387.04	**28,024.11**	0.102	0.624	0.206	3,472.33
KNN	24,484.12	27,118.17	0.102	0.633	0.210	3,389.17
RF	25,251.16	25,519.30	0.104	0.619	0.209	3,131.96
SVM	11,243.09	12,121.16	0.103	0.639	0.215	1,473.80
ContentBased	10,625.94	12,895.62	0.097	0.626	0.199	1,508.62
Random	14,254.11	12,861.34	0.093	0.645	0.200	1,560.50
Popularity	13,001.76	11,407.33	0.095	0.634	0.197	1,351.37

Table 5: Checkpoint 2 Sequential Modeling Performance Results

Method	Train Revenue	Test Revenue	Test Precision@K	Test NDCG@K	Test MRR	Test Disc. Revenue
RNN	22,121.17	**28,157.54**	0.101	0.624	0.203	3,331.25
AutoRegressive	23,467.05	23,601.27	0.104	0.619	0.209	2,852.88
Transformer	19,896.42	22,529.11	0.102	0.633	0.210	2,768.51
Popularity	12,470.18	11,936.98	0.093	0.645	0.200	1,526.43
ContentBased	11,916.62	11,276.26	0.087	0.632	0.185	1,337.92
Random	12,994.74	10,957.01	0.095	0.642	0.201	1,333.07

Table 6: Checkpoint 3 Graph-Based Recommendation Performance Results

Method	Train Revenue	Test Revenue	Test Precision@K	Test NDCG@K	Test MRR	Test Disc. Revenue
EfficientGraph	25,001.89	**25,787.36**	0.100	0.627	0.206	3,139.53
LightningGraph	24,019.84	23,545.65	0.101	0.602	0.194	2,756.92
SVM	12,303.94	13,208.32	0.101	0.604	0.195	1,517.50
Random	12,987.98	12,042.94	0.094	0.643	0.202	1,458.31
Popularity	14,233.50	11,782.54	0.095	0.636	0.199	1,344.79
ContentBased	12,085.54	11,486.34	0.094	0.625	0.193	1,402.50

Key Insights:

Simple beats complex: Gradient Boosting outperformed RNNs, Transformers, and Graph Neural Nets in *discounted revenue*

Revenue ≠ Ranking:

- Precision@K had *weak correlation* with revenue
- NDCG@K and MRR had *stronger correlations*, but still not optimal

Overfitting was common in deep models (especially Transformer, RNN) — mitigated with simplification & dropout

Temporal modeling (RNN) improved performance but not enough to beat well-tuned traditional models

Graph models provided stable, interpretable performance, but couldn’t beat XGBoost

Price and user behavior were critical features (e.g., item price, user avg purchase, popularity)

Final Strategy and Winning Approach

Winning Model: Gradient Boosting (XGBoost), reached 2831.22 discounted revenue (ranked 1st place on leaderboard)

Scoring Function: Expected Value = P(purchase) × Item Price

Goal: Maximize discounted revenue by ranking top-k items by expected value

Tuning Strategy: Grid search + manual optimization

```
XGBoost: n_estimators=100, learning_rate=0.1, max_depth=5,
         subsample=0.8, colsample_bytree=0.8,
         early_stopping_rounds=10, eval_metric="logloss"
```

Lessons Learned and Course Connections

Key Lessons Learned

- Traditional ML ≠ obsolete: XGBoost outperformed deep models like RNNs
- Complex ≠ better: Simpler, well-tuned models often delivered higher revenue
- Metric alignment matters: Optimizing ranking metrics alone does not maximize business goals
- Evaluation frameworks must match objectives (e.g., discounted revenue > Precision@K)

CS145 Concepts Applied

- Supervised Learning: Classification + regression models (XGBoost, RF)
- Unsupervised Learning: NMF, SVD (for graph-based embeddings)
- Neural Networks: RNNs, Transformers, GNNs for advanced modeling
- Model Selection & Tuning: Grid search, dropout tuning, regularization
- Metric Analysis: Correlation between ranking metrics & revenue

Further Thoughts:

Future work includes hybrid ensemble approaches combining traditional ML with sequential/graph methods, advanced temporal feature engineering, and multi-objective optimization frameworks incorporating revenue, user satisfaction, and fairness metrics. Our work demonstrates that successful data mining applications require careful problem formulation, systematic experimental design, and critical evaluation - core principles emphasized throughout CS145.

Thank you!