H & M Recommendation System

Ty Painter
Logan King
Connor Mignone

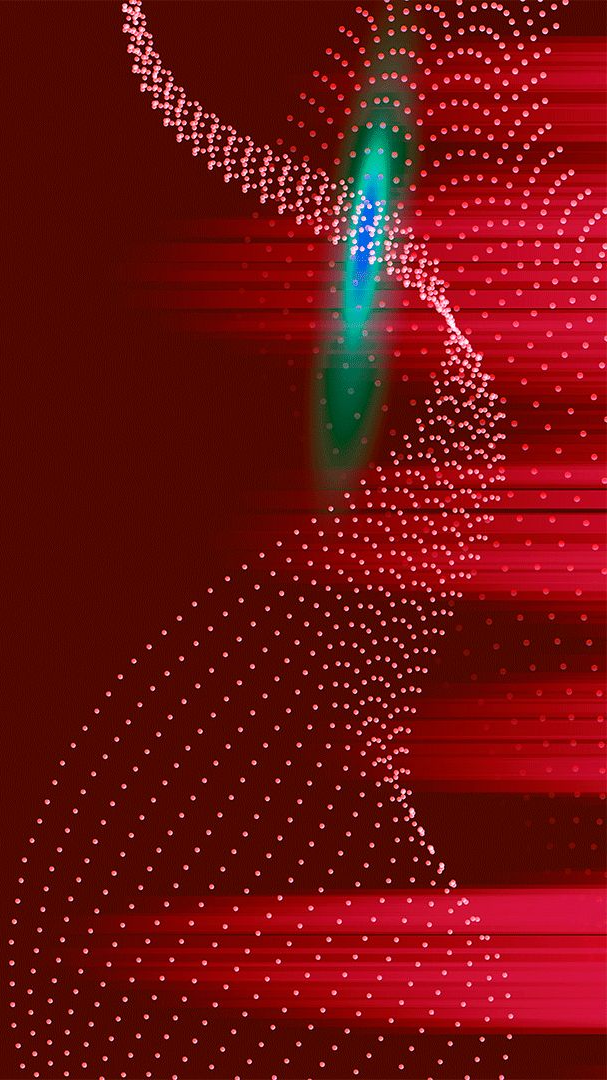
Are you a fashionista?

Recommendation algorithms can help!

Consumers have preferences for which articles of clothing that they will buy. We can use consumer buying habits to recommend items which consumers are most likely to purchase.

- Graph Neural Networks
- Collaborative Filtering

Improves consumer shopping experience and bottom line of H&M



The H&M Challenge

Predict the 12 most relevant items a consumer is likely to purchase within a given time.

- Dataset: 3 data sets
  a. Customer Information
  b. Item Information
  c. Transactions
- Evaluation: Scored with the mean average precision of 12 predictions
- Caveat: Bad recommendations are not penalized

Challenges?

Changing Plans

TorchRec vs. Explainable documented methods

Preprocessing

Dealing with features, embeddings, and formatting data

Model Framework

Tradeoffs between simplicity, performance, and scalability

Data Description

Size

31 million rows of transactions

Features

Customer article interaction

Graph

Heterogeneous Bipartite Graph

Framework

Subset Data

Subset observations and columns of interest

Structure data for modeling

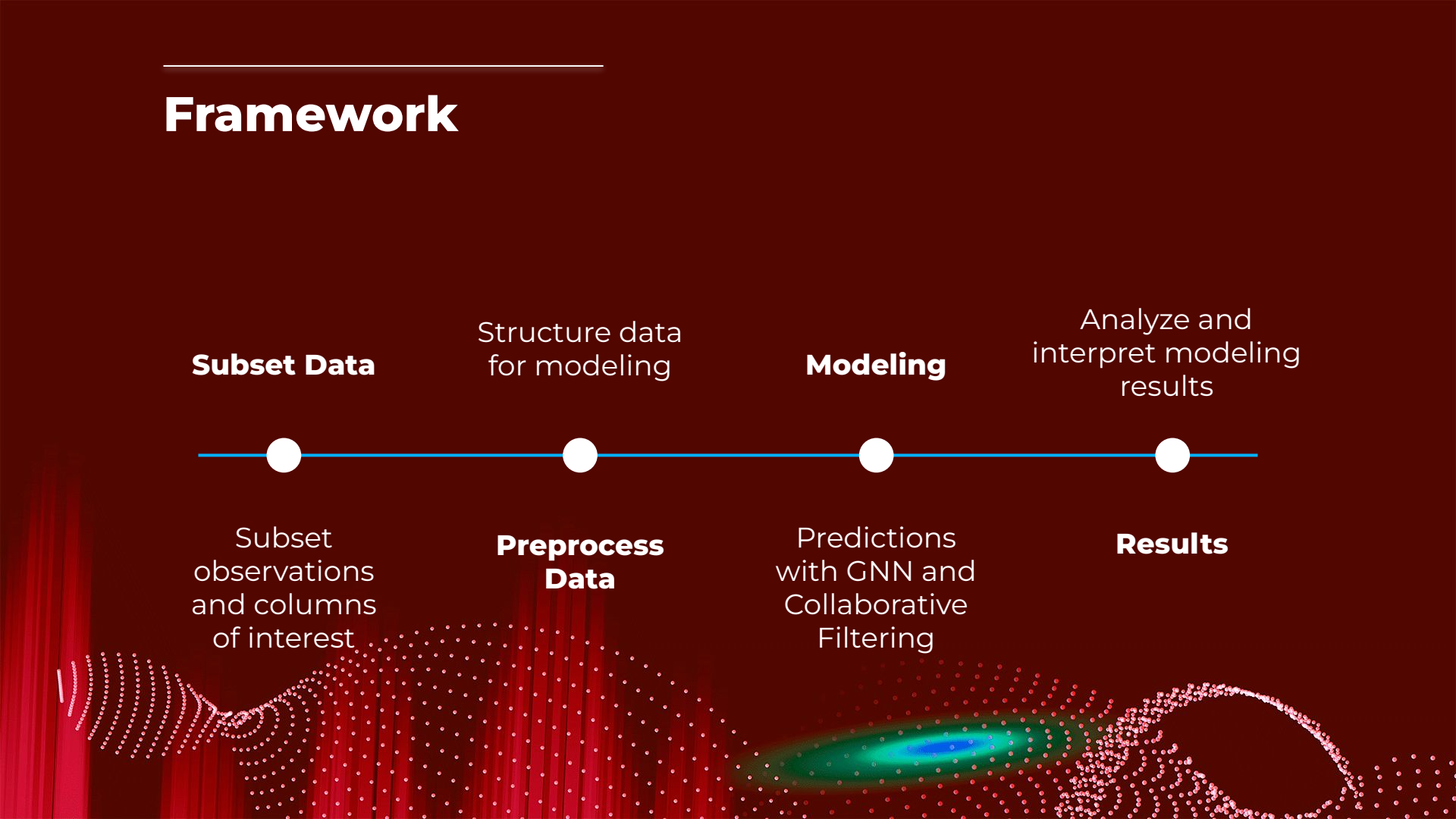
Preprocess Data

Modeling

Predictions with GNN and Collaborative Filtering

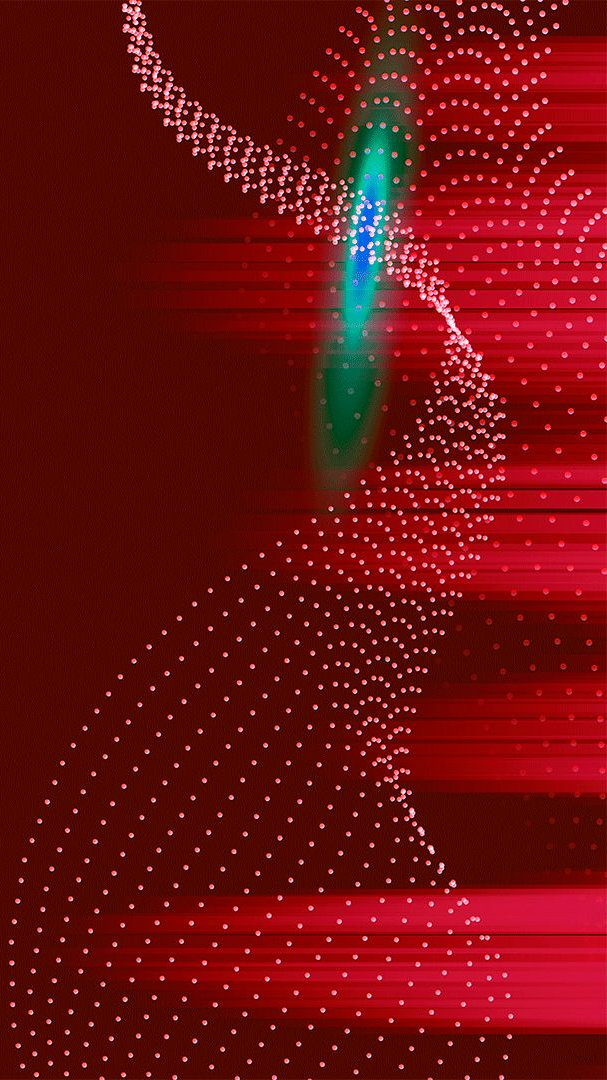
Analyze and interpret modeling results

Results



Subset Data

- Rows
  - ~30M
  - 20,000
- Columns
  - Customer ID
  - Article ID
  - ... future
- Train/Test split
  - 70/30
  - Training set
    - 5,099 unique customers
    - 6,134 unique articles



Data Preprocessing

Create unique numeric IDs

Label Encoding

All possible combinations of article and customer

Data Loader

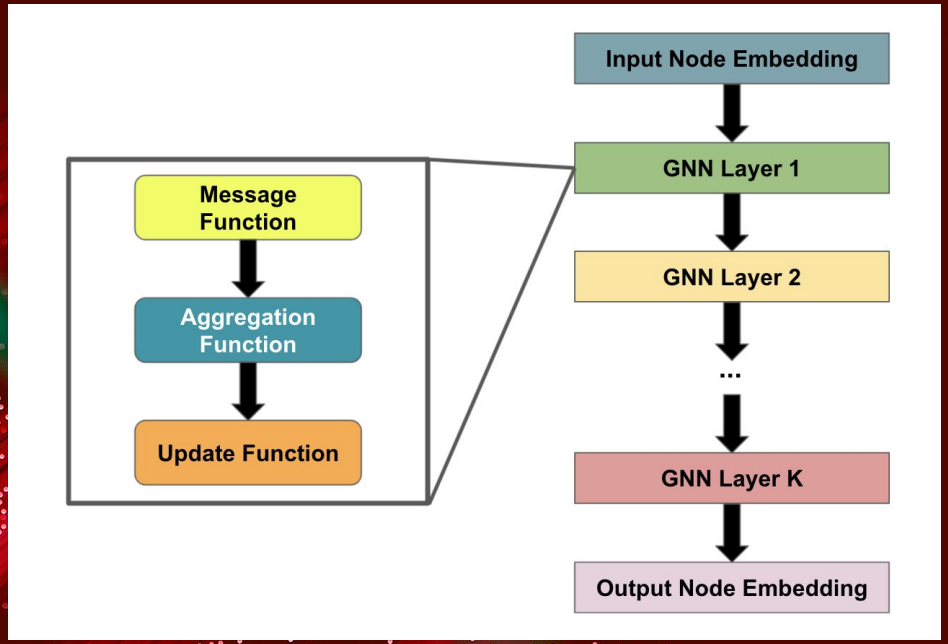
Unique index for bipartite, heterogeneous graph

Indexing

Test set must intersect with train set

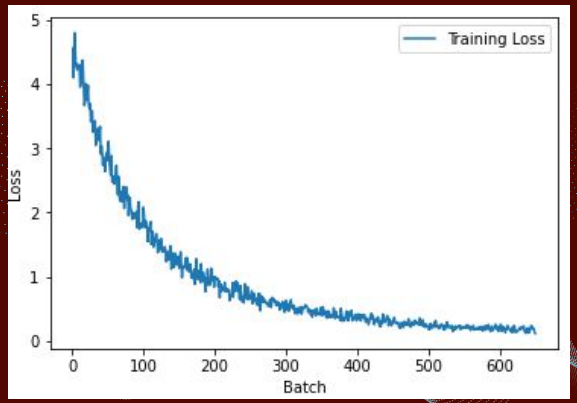
Cold Start

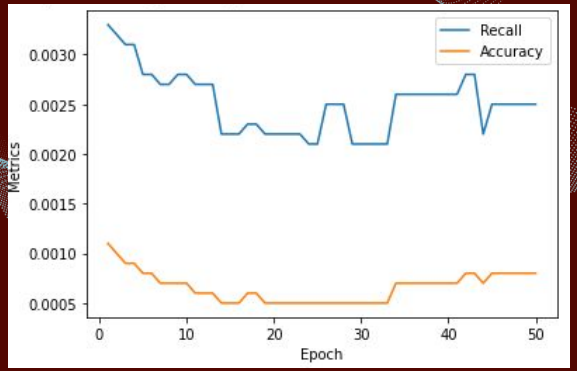
GNN with Collaborative Filtering



- Latent Features → Embeddings
  - Similarity and structure
- Message Propagation
- Aggregation
- Update
- Collaborative Filtering
  - Similar interests → Common products

Results and Implications





- Adam Optimizer
  - Weight Decay: 0.001
  - Learning Rate: 0.01
  - Drop: 0.01
- 50 Epochs
- Softplus loss function with a Beta of 1.5

User-Based Collaborative Filtering

- Ran on CPU
- Small subset for testing
    - Negative Sampling Expansion
- Using PyTorch, Fastai, and Recommenders

Epoch	Train Loss	Time
0	.006078	00:08
1	.005332	00:08
2	.005265	00:08
3	.003282	00:08
4	.00307	00:08
		Total:44.3 Seconds

Collaborative Filtering Results

Metrics	
Precision of K Predictions	24.51%
Recall of K Predictions	2.08%
RMSE	0.106232
MAE	0.013038

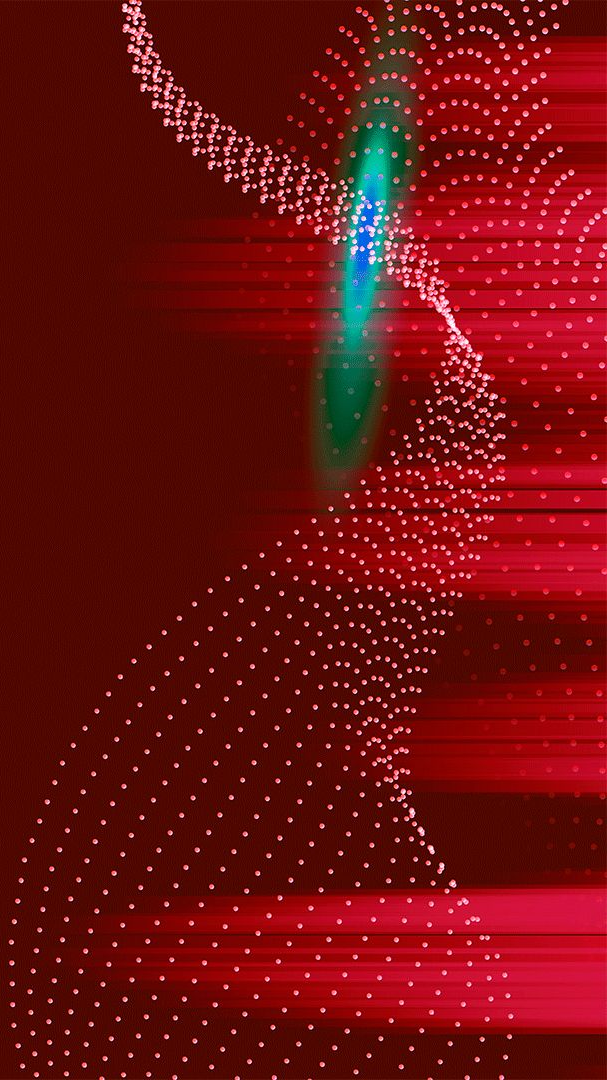
Conclusions / Next Steps

Conclusions:

- Model is training
- Poor accuracy
- Slightly better recall
- Better than no information rate
- More efficient tweaking, runtime Issues

Next Steps:

- Tweaking Neural Network
- Adding features of articles and customers
- Scaling collaborative filter



Thank You!