# MET CS 777 – Term Project Report

**Project:** CityEats Recommender (PySpark ALS)

**Student:** Srivatsav Shrikanth (sri99-svg)

**Date:** 10/13/25

## 1. Introduction

We build a top-N recommendation system using Alternating Least Squares (ALS) on explicit ratings. The goal is to generate quality recommendations at scale and evaluate ranking performance (Precision@K, Recall@K, NDCG@K).

## 2. Data

Source: Yelp-style explicit ratings processed into Parquet.

Schema: user\_id, item\_id, rating (double).

Location: data/silver\_explicit.parquet (or external link in DATA.md).

## 3. Methodology

• Index users/items with StringIndexer to integer ids (user\_idx, biz\_idx).

• Train/test split: global random split (seed 42).

• ALS hyperparameters: rank, regParam, maxIter, nonnegative, coldStartStrategy=drop.

• Evaluation: Precision@K, Recall@K, NDCG@K (positive threshold from config).

## 4. Implementation

• Entry script: jobs/train\_als\_local.py.

• Config: conf/config.yaml.

• Spark UI available on http://localhost:4040 during runs.

• Repro commands and environment are in README.md.

## 5. Results

|  |  |
| --- | --- |
| K | 50 |
| Precision@K | 0.0011 |
| Recall@K | 0.0129 |
| NDCG@K | 0.0042 |

Observations:

## • Metrics are computed with exclude-seen evaluation on a global split, which is stricter and typically yields lower values than include-seen baselines.

## • Runtime and stability improved after reducing shuffle partitions and ALS block counts; quality is expected to rise with minimum-interaction filtering and light hyperparameter tuning.

## 6. Discussion

## Hyperparameters (rank/regParam/maxIter) trade off quality vs. runtime; nonnegative factors and coldStart=drop avoid invalid predictions. Exclude-seen evaluation penalizes popular-item bias but reflects realistic serving. On local hardware, reducing shuffle partitions and ALS block counts stabilizes memory at the cost of some parallelism.

## 7. Conclusion & Future Work

## Current pipeline is reproducible and scalable. Next steps: add implicit-feedback experiments, enforce minimum interactions per user/item, run a small hyperparameter sweep, and scale training on Dataproc/Databricks; package a batch export or simple serving API.

## 8. How to Run

See README.md for spark-submit commands and parameter notes.

## 9. References

PySpark ALS documentation; Yelp academic dataset.