

9. Model Training and Evaluation

9.1 Training and Evaluation Scripts

Model training, evaluation, and inference are implemented as **separate, modular scripts** to reflect production-grade machine learning workflows.

Training Script



Location:

`src/models/train_model.py`

Responsibilities:

- Reads user-item interaction features from the PostgreSQL feature store
 - Trains a collaborative filtering model using **Matrix Factorization (SVD)**
 - Stores the trained model as a serialized artifact
 - Logs training parameters and artifacts using MLflow
-

Evaluation Script



Location:

`src/models/evaluate_model.py`

Responsibilities:

- Loads the trained model artifact
- Evaluates the model using ranking-based metrics:

- Precision@K
 - Recall@K
 - Logs evaluation metrics to MLflow for experiment tracking
-

Inference Script

 **Location:**

`src/models/inference.py`

Responsibilities:

- Loads the trained recommendation model
- Generates top-K product recommendations for a given user
- Demonstrates inference-time feature consumption

```
● (.venv) PS D:\BITS\DM4ML\DM4ML-Assignment-1> python -m src.models.train_model
2026/01/20 13:57:44 INFO mlflow.store.db.utils: Creating initial MLflow database tables...
2026/01/20 13:57:44 INFO mlflow.store.db.utils: Updating database tables
2026/01/20 13:57:44 INFO alembic.runtime.migration: Context impl SQLiteImpl.
2026/01/20 13:57:44 INFO alembic.runtime.migration: Will assume non-transactional DDL.
2026/01/20 13:57:44 INFO alembic.runtime.migration: Context impl SQLiteImpl.
2026/01/20 13:57:44 INFO alembic.runtime.migration: Will assume non-transactional DDL.
Model trained and saved successfully
● (.venv) PS D:\BITS\DM4ML\DM4ML-Assignment-1> python -m src.models.evaluate_model
2026/01/20 13:57:54 INFO mlflow.store.db.utils: Creating initial MLflow database tables...
2026/01/20 13:57:54 INFO mlflow.store.db.utils: Updating database tables
2026/01/20 13:57:54 INFO alembic.runtime.migration: Context impl SQLiteImpl.
2026/01/20 13:57:54 INFO alembic.runtime.migration: Will assume non-transactional DDL.
2026/01/20 13:57:54 INFO alembic.runtime.migration: Context impl SQLiteImpl.
2026/01/20 13:57:54 INFO alembic.runtime.migration: Will assume non-transactional DDL.
Precision@5: 0.1900
Recall@5: 0.9500
● (.venv) PS D:\BITS\DM4ML\DM4ML-Assignment-1> python -m src.models.inference
Recommendations for U1:
[ 2  1  3  7 11]
↖ (.venv) PS D:\BITS\DM4ML\DM4ML-Assignment-1>
```

9.2 Model Performance Report

Model performance is evaluated using **standard recommendation system metrics** that measure ranking quality.

Evaluation Metrics

- **Precision@5:** Measures the relevance of the top-5 recommended items
- **Recall@5:** Measures the coverage of relevant items in the top-5 recommendations

Performance Summary

A model performance report is generated based on the logged evaluation metrics and includes:

- Model type and configuration
- Training data source (PostgreSQL feature store)
- Precision@K and Recall@K scores
- Observations on model behavior and performance

These results are reproducible and traceable through MLflow experiment logs.

9.3 Tracked Model Metadata

All model-related metadata is tracked using **MLflow**, ensuring reproducibility and auditability.

Tracked Information

- **Run IDs:** Unique identifiers for each experiment run
- **Parameters:** Model type, latent dimensions, data source
- **Metrics:** Precision@5, Recall@5

- **Artifacts:** Serialized trained model file

Model Artifact

`models/svd_model.pkl`

Experiment Tracking

MLflow provides a centralized UI to compare runs and inspect model performance:

`mlflow ui`

The image shows two screenshots of the MLflow UI.

Top Screenshot (Home Page):

- Header:** mlflow 3.8.1, 127.0.0.1:5000
- Left Sidebar:**
 - + New
 - Home
 - Experiments
 - Models
 - Prompts
- Information about UI telemetry:** MLflow collects usage data to improve the product. To confirm your preferences, please visit the settings page in the navigation sidebar. To learn more about what data is collected, please visit the documentation.
- Get started:**
 - Log traces: Trace LLM applications for debugging and monitoring.
 - Run evaluation: Iterate on quality with offline evaluations and comparisons.
 - Train models: Track experiments, parameters, and metrics throughout training.
 - Register prompts: Manage prompt updates and collaborate across teams.
- Experiments:**

Name	Time created	Last modified	Description	Tags
Default	01/20/2026, 01:57:11 PM	01/20/2026, 01:57:11 PM	-	
- Discover new features:**
 - MLflow MCP server: Connect your coding assistants and AI applications to MLflow and automatically analyze your experiments and traces.
 - Optimize prompts: Access the state-of-the-art prompt optimization algorithms such as MPROQ2, GPTA, through MLflow Prompt Registry.
 - Agents-as-a-judge: Leverage agents as a judge to perform deep trace analysis and improve your evaluation accuracy.
 - Dataset tracking: Track dataset lineage and versions and effectively drive the quality improvement loop.

Bottom Screenshot (Machine Learning Run View):

- Header:** mlflow 3.8.1, Default Machine learning
- Left Sidebar:**
 - Runs
 - Models
 - Traces
- Search Bar:** metrics.mse < 1 and params.model = "tree"
- Run List:**
 - Run Name
 - capable-basis-319 (red dot)
 - bulky-worm-860 (orange dot)
 - furry-pug-185 (pink dot)
- Metrics:**
 - Model metrics (2): precision_at_5 (0.19), recall_at_5 (0.95)
 - System metrics (0):
- Notes:** 3 matching runs

9.4 Summary

The model training and evaluation stage implements a complete and reproducible workflow for recommendation systems. Training, evaluation, and inference are clearly separated, performance metrics are computed using standard measures, and all model metadata is tracked using MLflow. This ensures transparency, repeatability, and alignment with modern MLOps best practices.