Machine Learning Capstone
Personalized Course Recommender Systems

Roberto Villafuerte Carrillo

IBM ML321EN Capstone

October 6, 2025

Problem & Goals

Objective. Build a recommender system that suggests relevant online courses to learners.

Key questions

- What content patterns exist in course catalog & enrollments?
- How do content-based vs. collaborative filtering (CF) approaches compare?
- Can neural embeddings improve recommendations?

Deliverables

- EDA and feature engineering
- Content-based recommenders (3 variants)
- CF: KNN, NMF, NN-embeddings
- Supervised models (Regression & Classification using embeddings)
- Offline evaluation & conclusions

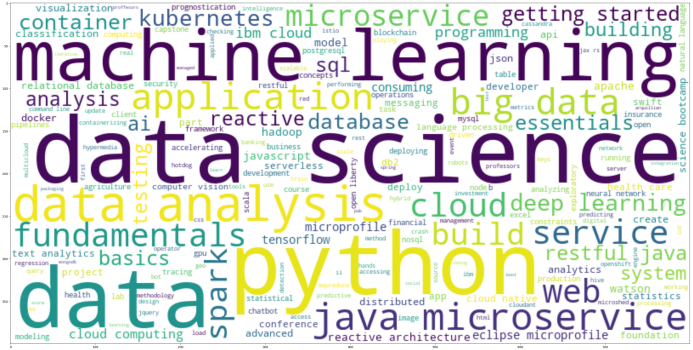
Data Overview

- **Ratings / Enrollments:** 233,306
- **Unique Users:** 33,901
- **Courses (with metadata):** 307
- **Rating values:** $\{3, 4, 5\}$

User–Item Matrix

$$33{,}901 \times 307 = 10{,}407{,}607 \quad \Rightarrow \quad \text{Density } \approx 2.24\%, \ \text{Sparsity } \approx 97.76\%$$

Course Titles — Keyword Snapshot



¡/div¿

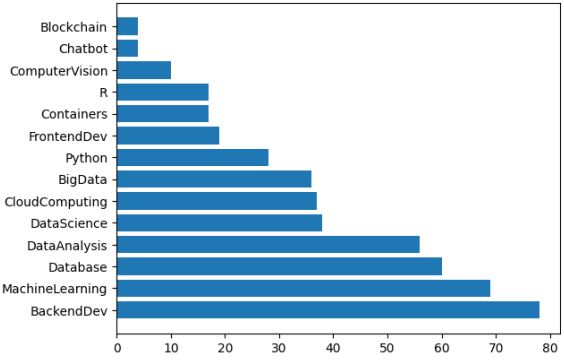
Prominent themes

- Python, Data Science, Machine Learning, Big Data, AI
- TensorFlow, Spark, SQL, Visualization
- Cloud, Containers/Docker, Microservices

User Activity — Courses per User

- **Users**: 33,901 **Mean**: 6.88 **Median**: 6
- **Min**: 1 **75th pct**: 9 **Max**: 61

Long–tail engagement: most users rate a handful; a small cohort is highly active.



Top–20 Most Enrolled Courses (63.3% of all enrollments)

Course ID	Title	Enrolls
PY0101EN	python for data science	14,936
DS0101EN	introduction to data science	14,477
BD0101EN	big data 101	13,291
BD0111EN	hadoop 101	10,599
DA0101EN	data analysis with python	8,303
DS0103EN	data science methodology	7,719
ML0101ENv3	machine learning with python	7,644
BD0211EN	spark fundamentals i	7,551
DS0105EN	data science hands on with open source tools	7,199
BC0101EN	blockchain essentials	6,719
DV0101EN	data visualization with python	6,709
ML0115EN	deep learning 101	6,323
CB0103EN	build your own chatbot	5,512
RP0101EN	r for data science	5,237
ST0101EN	statistics 101	5,015
CC0101EN	introduction to cloud	4,983
CO0101EN	docker essentials: a developer introduction	4,480

Feature Engineering — Summary

- **Content features**: binary genre flags, bag-of-words/TF–IDF from titles/descriptions (if available)
- **Interaction features**: user/item IDs, ratings (3/4/5)
- **Neural embeddings**: user/item embedding vectors (dim = 16), concatenation/aggregation for supervised tasks
- **User profiles**: genre-weighted profiles from historical enrollments/ratings

Content-Based: User Profile & Genres

Method

- Build a user vector as normalized average of genres from interacted courses (optionally weighted by rating)
- Score a candidate course by cosine similarity to the user profile

Output

- Top–N per user; cold-start friendly

Add: Insert a small example table (`reports/figures/cb_profile_example.png`)

Content-Based: Course–Course Similarity

Method

- Represent courses via genres + title TF–IDF vectors
- Recommend items similar to a user's previously liked items (item-item cosine)

Notes

- Good for "more like this"
- Diversify via MMR or genre constraints

Content-Based: User Profile Clustering

Method

- KMeans on user genre profiles to identify personas
- Recommend cluster-top courses to members

Diagnostics

- Silhouette / inertia plots for K selection

Collaborative Filtering: KNN (Surprise)

Setup

- KNNBasic / KNNWithMeans, user- or item-based
- Similarities: cosine / msd / pearson; typical $k \in [20, 80]$

Metrics

- RMSE/MAE on ratings; Precision@K / Recall@K for top–N

Collaborative Filtering: NMF

Method

$$A \approx U \cdot I^\top, \quad U \in \mathbb{R}_+^{|\mathcal{U}| \times r},\ I \in \mathbb{R}_+^{|\mathcal{I}| \times r}$$

- Rank r (e.g., 37 as used in lab); train on sparse matrix
- Predict $\hat{r}_{ui} = U_u \cdot I_i^\top$

Collaborative Filtering: Neural Embeddings

Architecture

- Embedding layers for user & item (dim $= 16$)
- Concatenate $\rightarrow$ MLP $\rightarrow$ regression head (rating)

Outputs

- Trained embeddings reusable for other tasks

Supervised: Regression on Interaction Embeddings

Features

- Aggregated interaction vector: $X = U\_e \oplus I\_e$ (element-wise add)

Models

- Linear Regression, Ridge/Lasso/ElasticNet

Metric

- RMSE on held-out test set

Supervised: Classification of Rating Mode

Labels

- Encode $\{3, 4, 5\} \rightarrow \{0, 1, 2\}$ via LabelEncoder

Models

- Logistic Regression, Random Forest, SVM, Gradient Boosting

Metrics

- Accuracy, Precision, Recall, F1

Evaluation Protocol

- **Split:** train/validation/test on interactions
- **Rating metrics:** RMSE/MAE (CF & regression)
- **Ranking metrics:** Precision@K, Recall@K, MAP@K, nDCG@K
- **Cold-start:** evaluate new users/items via content-based

Model Comparison — Top-10 (by F1)

Model	Aggregation	Accuracy	Precision	Recall	F1
SVC	concat	0.678500	0.689600	0.678500	0.682000
SVC	sum	0.676500	0.687200	0.676500	0.679800
RandomForest	sum	0.648700	0.651900	0.648800	0.650100
SVC	prod	0.634600	0.655400	0.634600	0.639900
RandomForest	concat	0.638100	0.636000	0.638200	0.636900
RandomForest	prod	0.630600	0.637600	0.630700	0.633300
LogisticRegression	prod	0.615900	0.613500	0.616000	0.614600
GradientBoosting	prod	0.615100	0.612600	0.615300	0.613800
GradientBoosting	concat	0.594000	0.583900	0.594300	0.583100
GradientBoosting	sum	0.570200	0.560100	0.570400	0.556600

Table: Top-10 models overall (by macro-F1) from the classification lab.

Best-by-aggregation: concat $\rightarrow$ SVC (F1 = 0.682) sum $\rightarrow$ SVC (F1 = 0.680) prod $\rightarrow$ SVC (F1 = 0.640).

Conclusions

- The catalog is **IT-focused** (Python/DS/ML/Cloud) with **head-heavy popularity** (Top–20 = 63.3%).
- **Content-based** methods handle cold-start and give explainable suggestions.
- **CF** captures collaborative signals; NMF and neural embeddings provide compact latent factors.
- **Supervised** models on embeddings are a flexible add-on for rating prediction.

Recommendations & Future Work

- Hybrid ranker (Content + CF) with learning-to-rank on offline labels
- Diversification/serendipity constraints to fight popularity bias
- Contextual bandits for online exploration vs. exploitation
- A/B testing framework; fairness & coverage monitoring

Environment & Reproducibility

- Python 3.12; numpy 1.26.4; pandas 2.3.2; seaborn 0.13.2; matplotlib 3.10.6
- wordcloud 1.9.4; scikit-learn; Surprise (for KNN/NMF)
- Repo layout with `data/`, `notebooks/`, `src/`, `reports/figures/`

Notes

- Save figures used in slides to `reports/figures/`
- Set seeds (`rs = 123`) for reproducibility

Thank you

Questions?