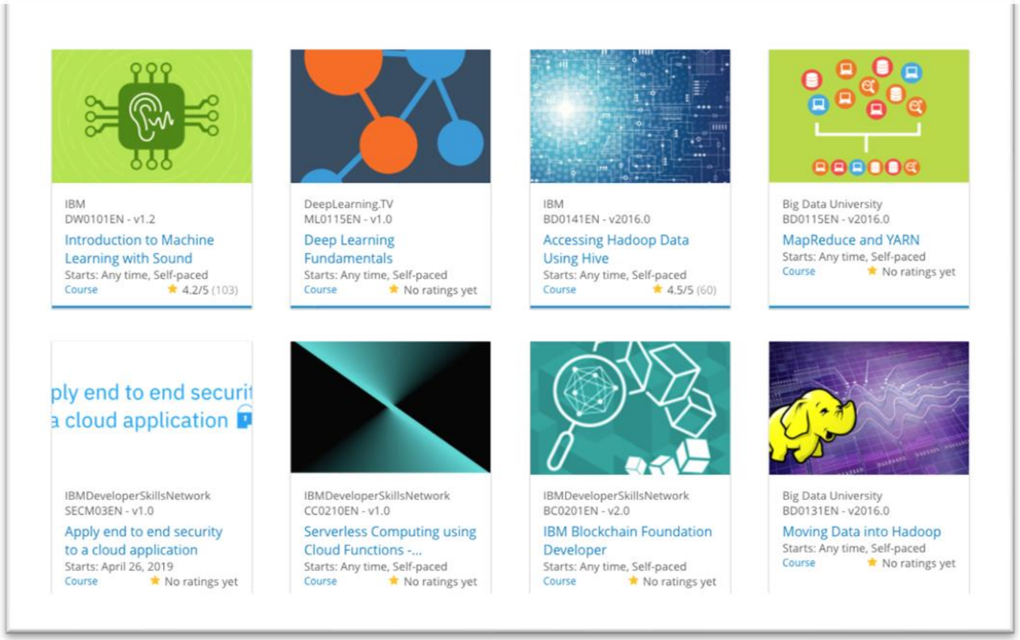
Online Course Recommender System with Machine Learning

Gustavo Rodrigues Alves Knudsen
2nd April, 2025



IBM
DW0101EN - v1.2
Introduction to Machine Learning with Sound
Starts: Any time, Self-paced
Course ★ 4.2/5 (103)

DeepLearning.TV
ML0115EN - v1.0
Deep Learning Fundamentals
Starts: Any time, Self-paced
Course ★ No ratings yet

IBM
BD0141EN - v2016.0
Accessing Hadoop Data Using Hive
Starts: Any time, Self-paced
Course ★ 4.5/5 (60)

Big Data University
BD0115EN - v2016.0
MapReduce and YARN
Starts: Any time, Self-paced
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
SECM03EN - v1.0
Apply end to end security to a cloud application
Starts: April 26, 2019
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
CC0210EN - v1.0
Serverless Computing using Cloud Functions -...
Starts: Any time, Self-paced
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
BC0201EN - v2.0
IBM Blockchain Foundation Developer
Starts: Any time, Self-paced
Course ★ No ratings yet

Big Data University
BD0131EN - v2016.0
Moving Data into Hadoop
Starts: Any time, Self-paced
Course ★ No ratings yet

Outline

- Introduction and Background
- Exploratory Data Analysis
- Content-based Recommender System using Unsupervised Learning
- Collaborative-filtering based Recommender System using Supervised learning
- Conclusion
- Appendix

Introduction

Overview

- The project was developing and evaluating different recommendation systems for an education institution by analysing user-item(course) interactions to predict preferences and course ratings. The dataset has over 31,000 users and 125 courses, containing many features describing the themes of each courses, user interests, and previous ratings by each student.
- Recommendation systems explored including: Content-Based Filtering, Collaborative Filtering (both user-based, item-based, using non-negative matrix factorization), Predictive models (neural networks, regression models, classification models).

Introduction

Objective

- Predict how users will rate courses they have not taken.
- Give personalized course recommendations tailored to user preferences.
- Handle new users and courses easily (scalable).
- Easy to interpret recommendations.

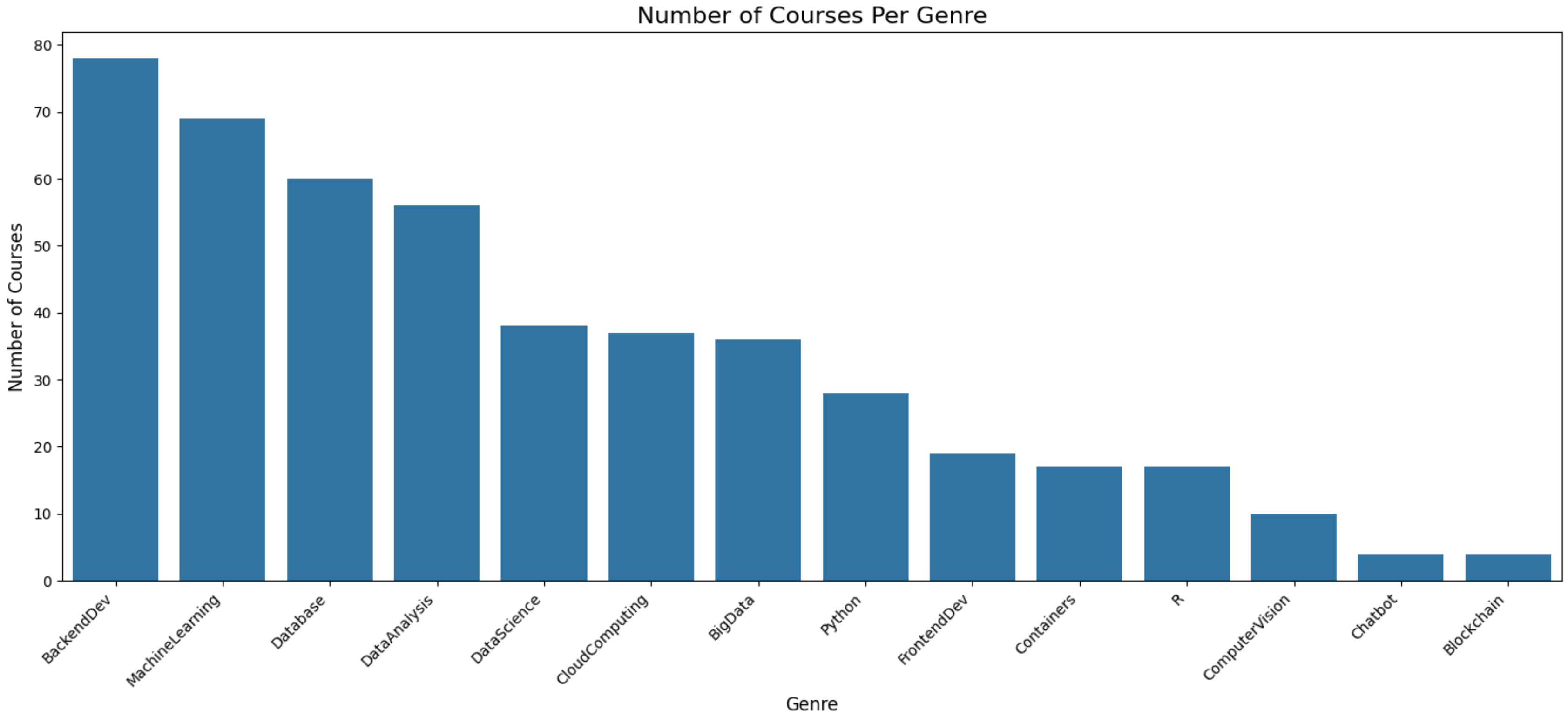
Hypotheses

- Collaborative Filtering will perform better than content-based methods.
- Classification models will predict the user ratings better than regression models.
- NMF will outperform KNN as methods of Collaborative Filtering.
- K-Means Clustering will be the best performing unsupervised clustering method for clustering-based recommender systems.
- The Neural Network will be the best performing predictor in terms of RMSE.

Exploratory Data Analysis

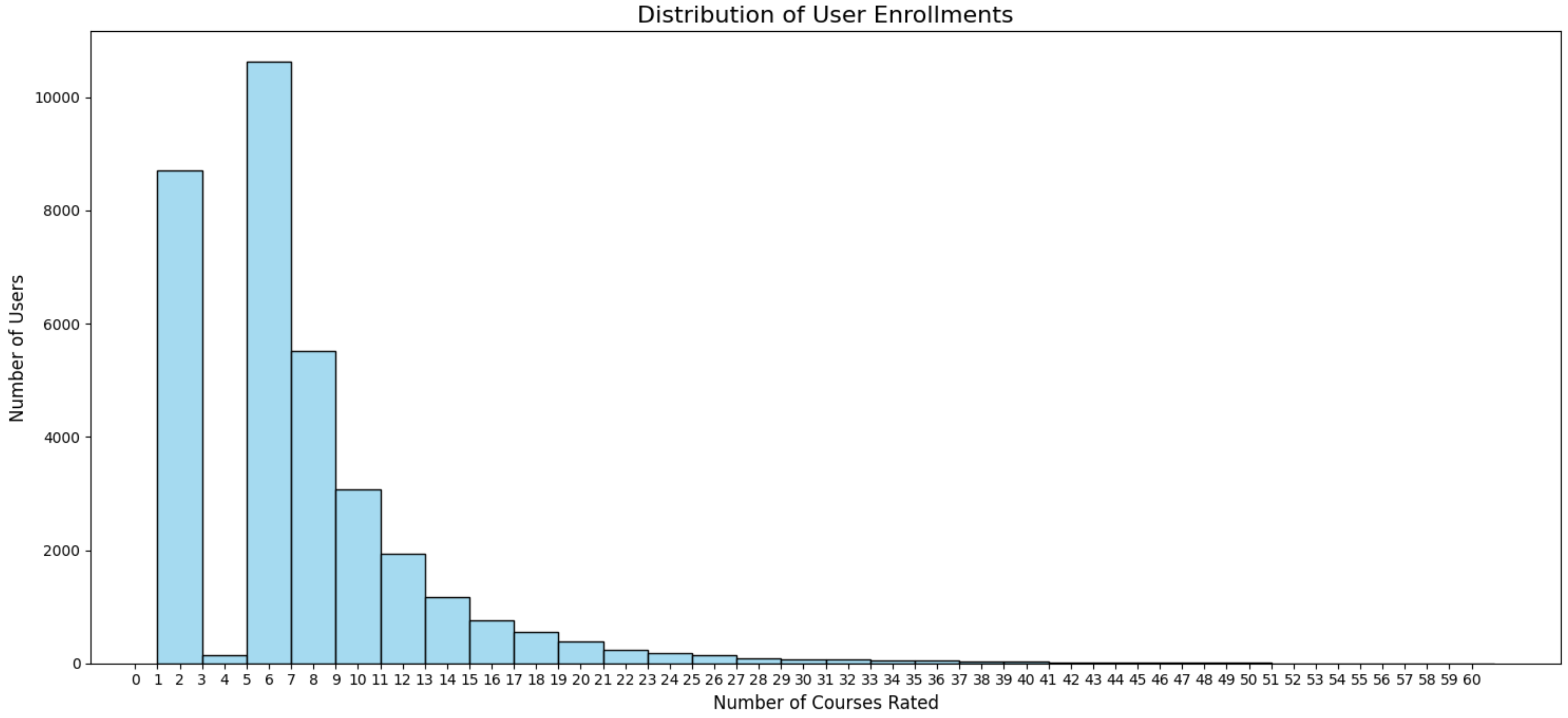
Course counts per genre

- Very few courses in Computer Vision, Chatbot, Blockchain. Especially compared to Backend Development, Machine Learning, DB, Data Analysis, etc.



Course enrollment distribution

- Histogram shows the distribution of how many people enrolled in how many classes. Most students took 1, 2, or 3 classes with a weird drop at 4. Most people took less than 9 classes, which means less data to base the predictions on.

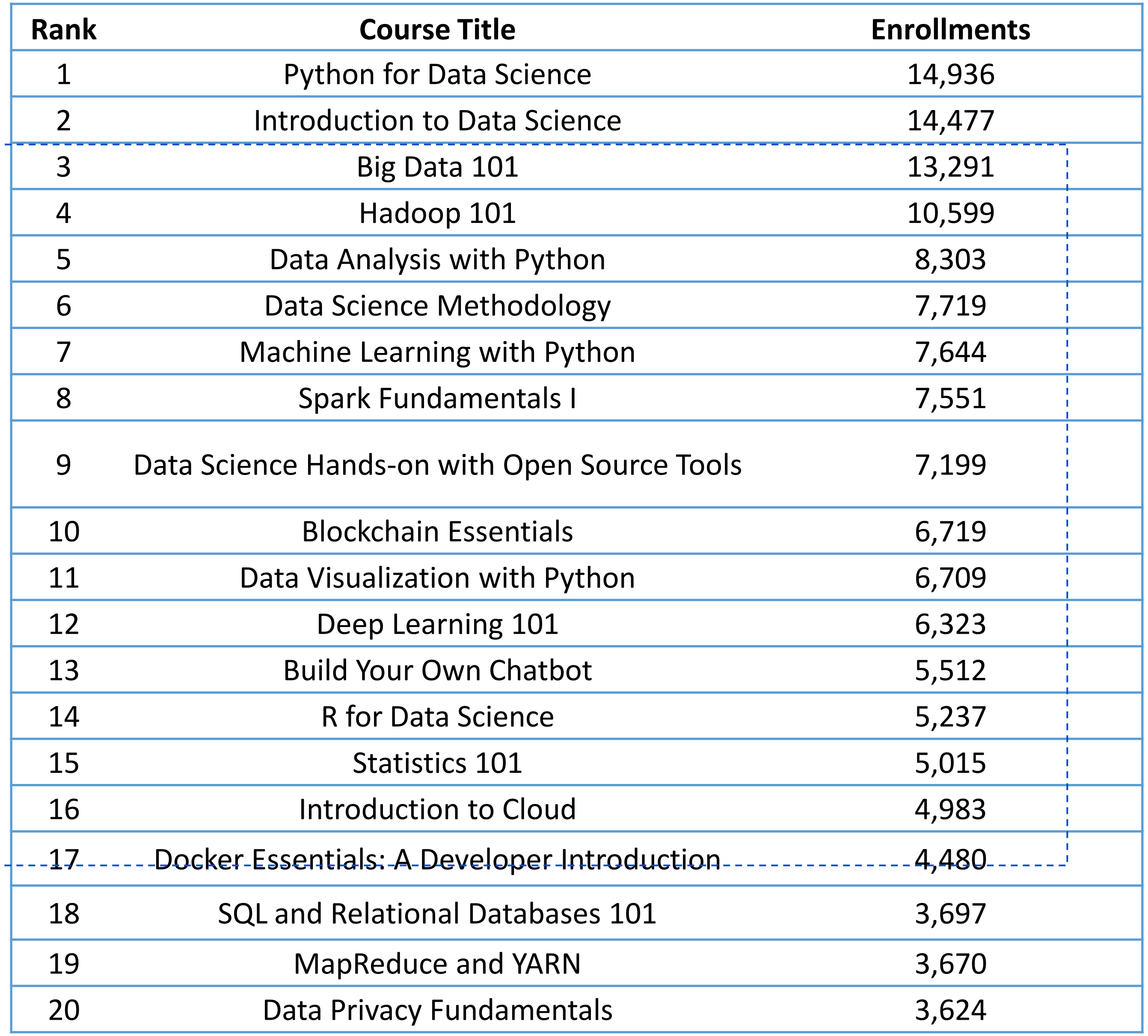


20 most popular courses

The most popular courses are:

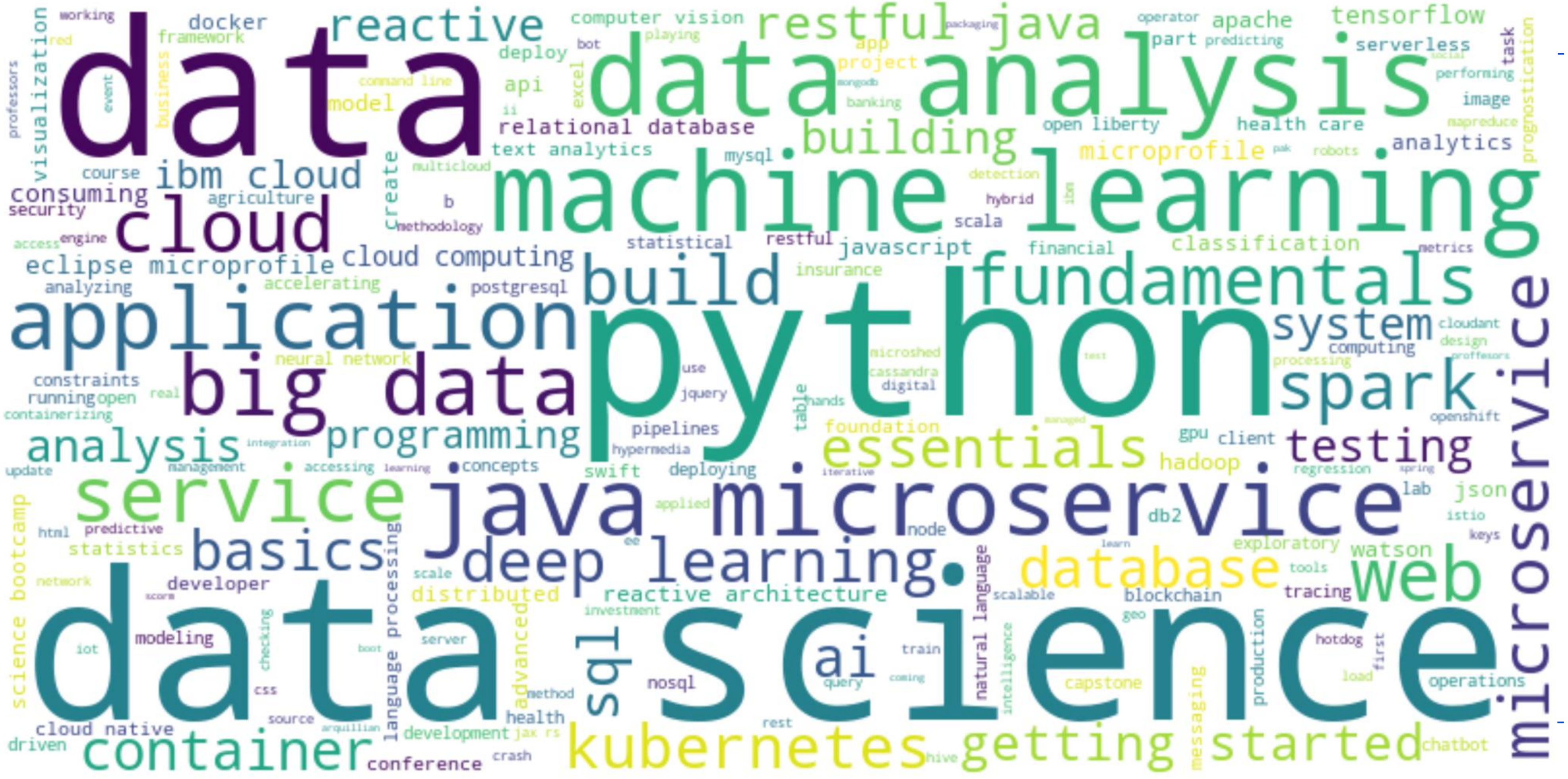
- Python for Data Science
- Introduction to Data Science
- Big Data 101
- Hadoop 101

Which are all above 10k enrollments.

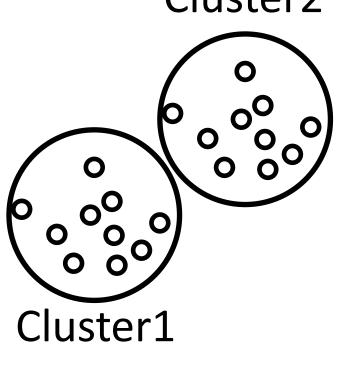


Rank	Course Title	Enrollments
1	Python for Data Science	14,936
2	Introduction to Data Science	14,477
3	Big Data 101	13,291
4	Hadoop 101	10,599
5	Data Analysis with Python	8,303
6	Data Science Methodology	7,719
7	Machine Learning with Python	7,644
8	Spark Fundamentals I	7,551
9	Data Science Hands-on with Open Source Tools	7,199
10	Blockchain Essentials	6,719
11	Data Visualization with Python	6,709
12	Deep Learning 101	6,323
13	Build Your Own Chatbot	5,512
14	R for Data Science	5,237
15	Statistics 101	5,015
16	Introduction to Cloud	4,983
17	Docker Essentials: A Developer Introduction	4,480
18	SQL and Relational Databases 101	3,697
19	MapReduce and YARN	3,670
20	Data Privacy Fundamentals	3,624

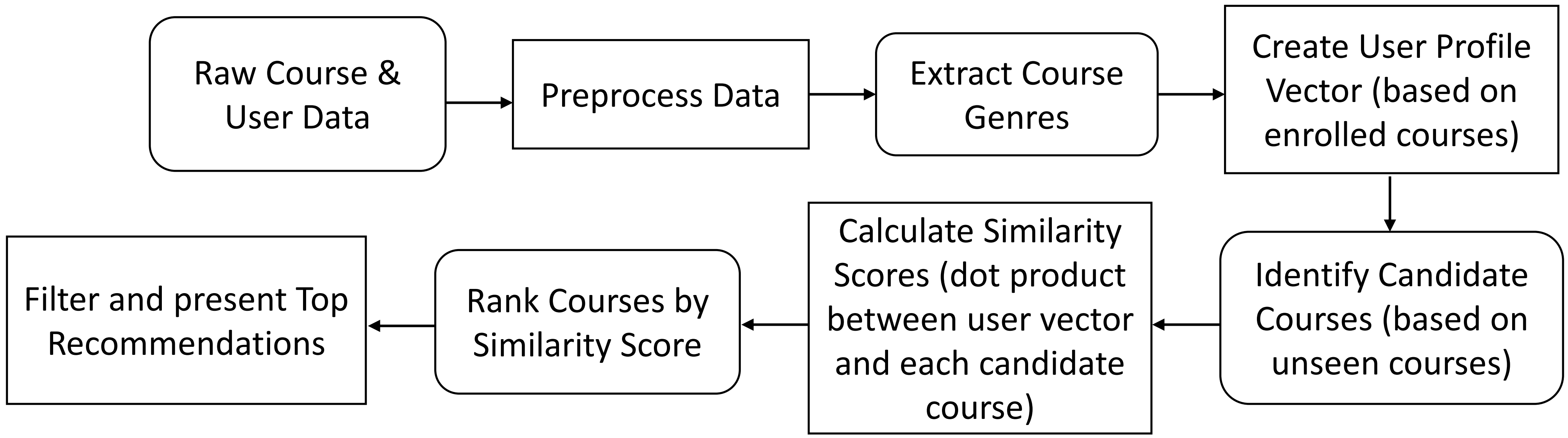
Word cloud of course titles



Content-based Recommender System using Unsupervised Learning



Flowchart of content-based recommender system using user profile and course genres


Raw Course & User Data
Preprocess Data
Extract Course Genres
Create User Profile Vector (based on enrolled courses)
Identify Candidate Courses (based on unseen courses)
Calculate Similarity Scores (dot product between user vector and each candidate course)
Rank Courses by Similarity Score
Filter and present Top Recommendations

Evaluation results of user profile-based recommender system

Score Threshold = 20.0 (lower gets too many, higher many people don't get any)

Sample of Recommendation Scores

	USER	COURSE_ID	SCORE
0	2	ML0201EN	43.0
1	2	GPXX0ZG0EN	43.0
2	2	GPXX0Z2PEN	37.0
3	2	DX0106EN	47.0
4	2	GPXX06RFEN	52.0
...	...	...	...
479126	2102680	GPXX04P5EN	23.0
479127	2102680	ML0101EN	29.0
479128	2102680	excourse21	29.0
479129	2102680	excourse22	29.0
479130	2102680	excourse49	20.0

Average recommendations per user: 28.94

Min recommendations for a user: 1

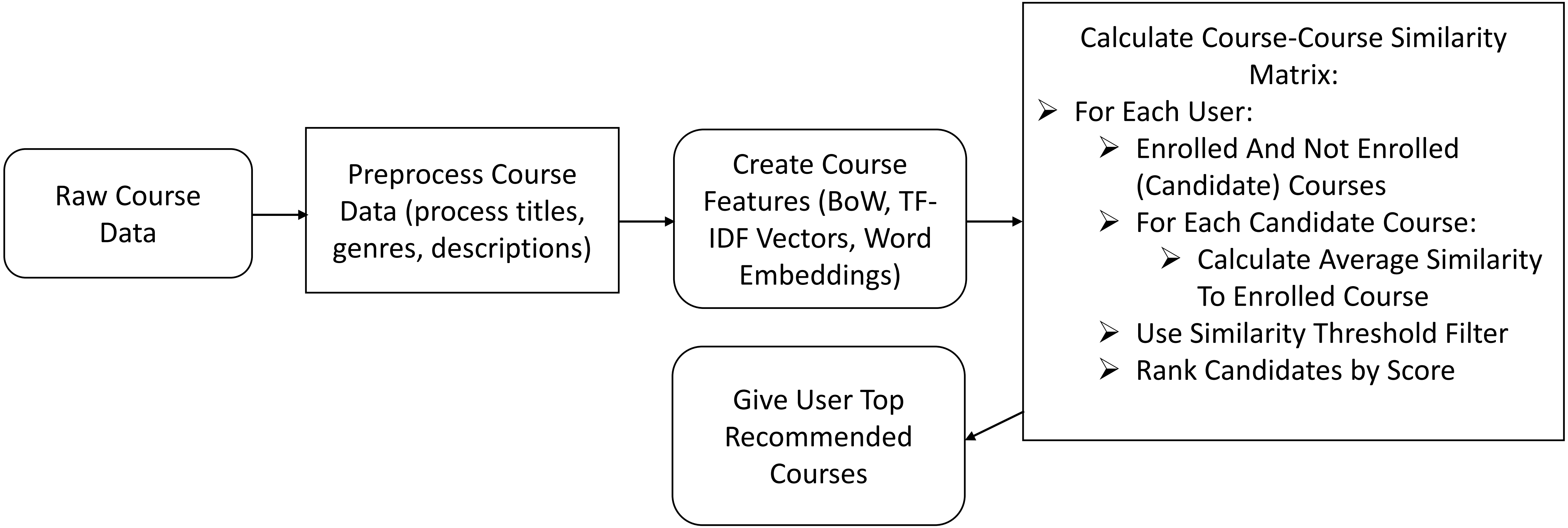
Max recommendations for a user: 236

Users with at least one recommendation: 16554 out of 33901 (48.8%)

Top 10 Most Frequently Recommended Courses:

1. foundations for big data analysis with sql
   Course ID: excourse72, Recommended 9138 times
2. analyzing big data with sql
   Course ID: excourse73, Recommended 9138 times
3. getting started with the data apache spark makers build
   Course ID: TMP0105EN, Recommended 8954 times
4. analyzing big data in r using apache spark
   Course ID: RP0105EN, Recommended 8769 times
5. spark overview for scala analytics
   Course ID: SC0103EN, Recommended 7970 times
6. cloud computing applications part 2 big data and applications in the cloud
   Course ID: excourse31, Recommended 7853 times
7. applied machine learning in python
   Course ID: excourse21, Recommended 7671 times
8. introduction to data science in python
   Course ID: excourse22, Recommended 7671 times
9. accelerating deep learning with gpu
   Course ID: ML0122EN, Recommended 7633 times
10. spark fundamentals ii
    Course ID: BD0212EN, Recommended 7203 times

Flowchart of content-based recommender system using course similarity



Raw Course Data

Preprocess Course Data (process titles, genres, descriptions)

Create Course Features (BoW, TF-IDF Vectors, Word Embeddings)

Calculate Course-Course Similarity Matrix:

- For Each User:
    - Enrolled And Not Enrolled (Candidate) Courses
    - For Each Candidate Course:
        - Calculate Average Similarity To Enrolled Course
    - Use Similarity Threshold Filter
    - Rank Candidates by Score

Give User Top Recommended Courses

Evaluation results of course similarity based recommender system

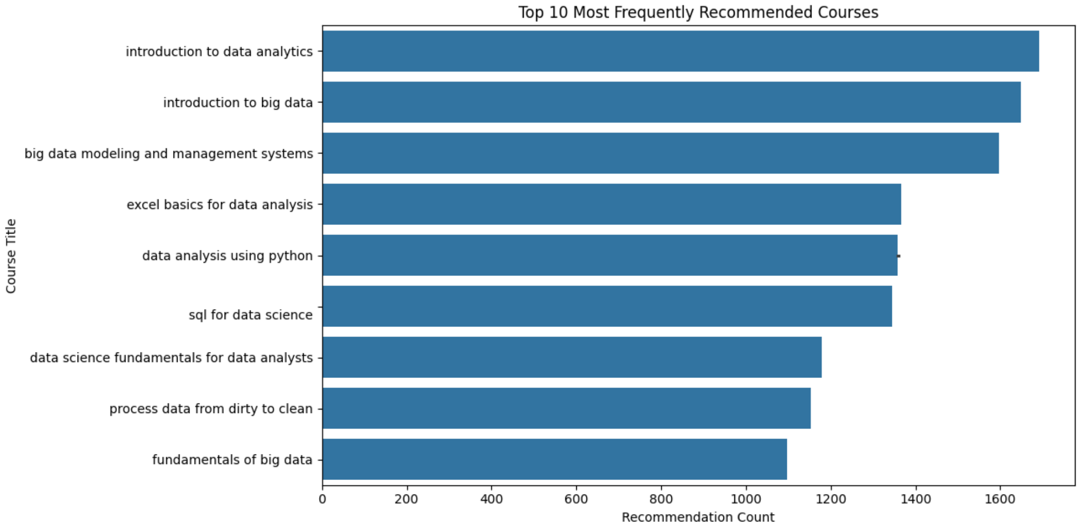
Threshold: 0.999 Embeddings, 0.5 BoW, 0.3 TF-IDF, 0.5 Downloaded Sim. Matrix

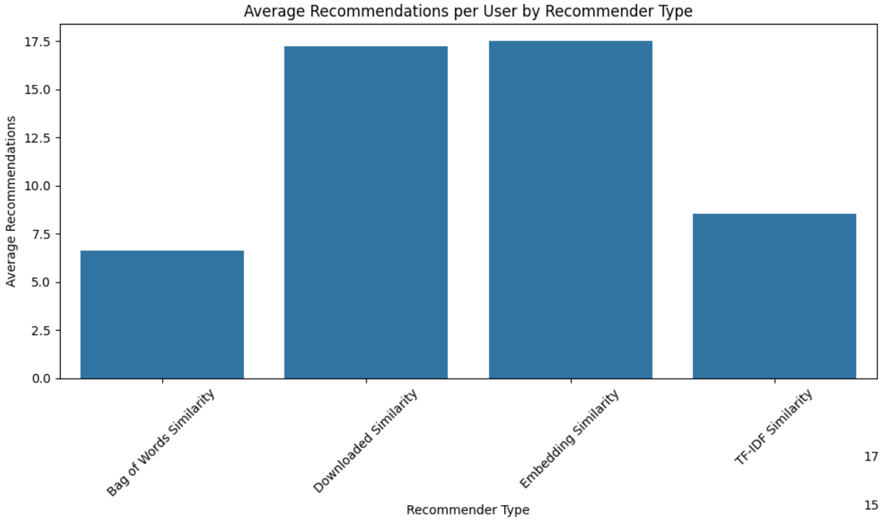
Recommender	Average Recommendations	Min Recommendations	Max Recommendations	Users With Recommendations
BoW	6.641598	1	50	851
Downloaded Sim. Matrix	17.244121	1	65	893
Embedding	17.506507	9	20	999
TF-IDF	8.556161	1	43	917

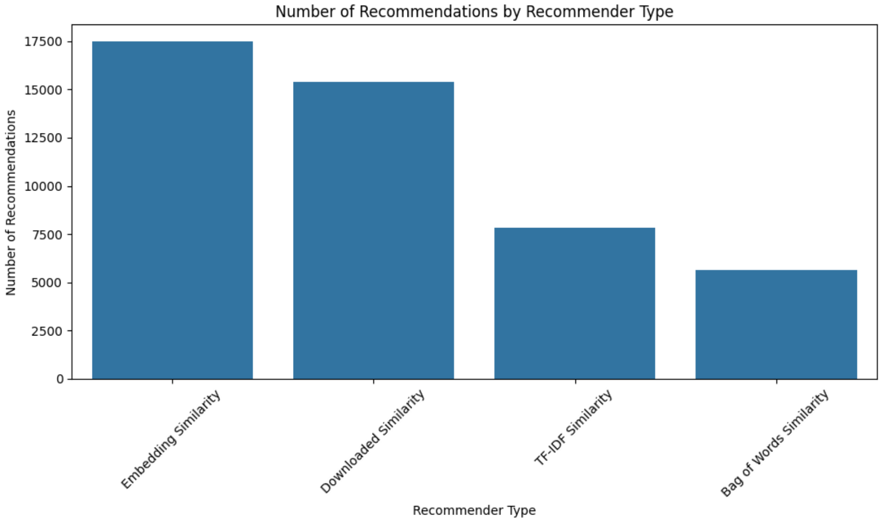
Top 10 Most Frequently Recommended Courses (Overall):

1. introduction to data analytics Course ID: excourse32, Recommended 1691 times
2. introduction to big data Course ID: excourse67, Recommended 1649 times
3. big data modeling and management systems Course ID: excourse68, Recommended 1597 times
4. excel basics for data analysis Course ID: excourse33, Recommended 1367 times
5. data analysis using python Course ID: excourse23, Recommended 1359 times
6. 6. data analysis using python Course ID: excourse36, Recommended 1357 times
7. sql for data science Course ID: excourse04, Recommended 1345 times
8. data science fundamentals for data analysts Course ID: excourse65, Recommended 1180 times
9. process data from dirty to clean Course ID: excourse09, Recommended 1153 times
10. fundamentals of big data Course ID: excourse74, Recommended 1096 times

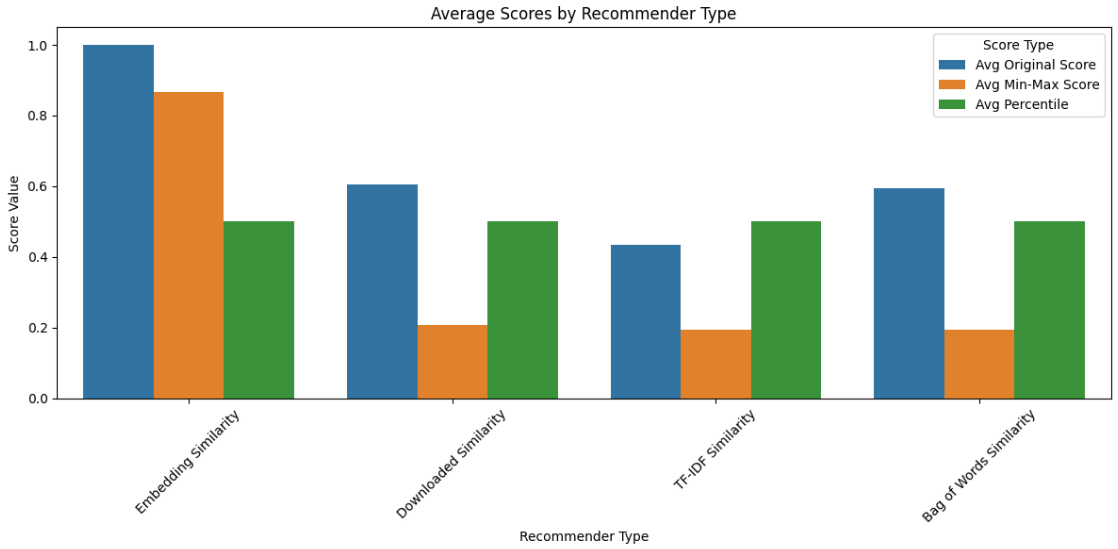
Evaluation results of course similarity based recommender system

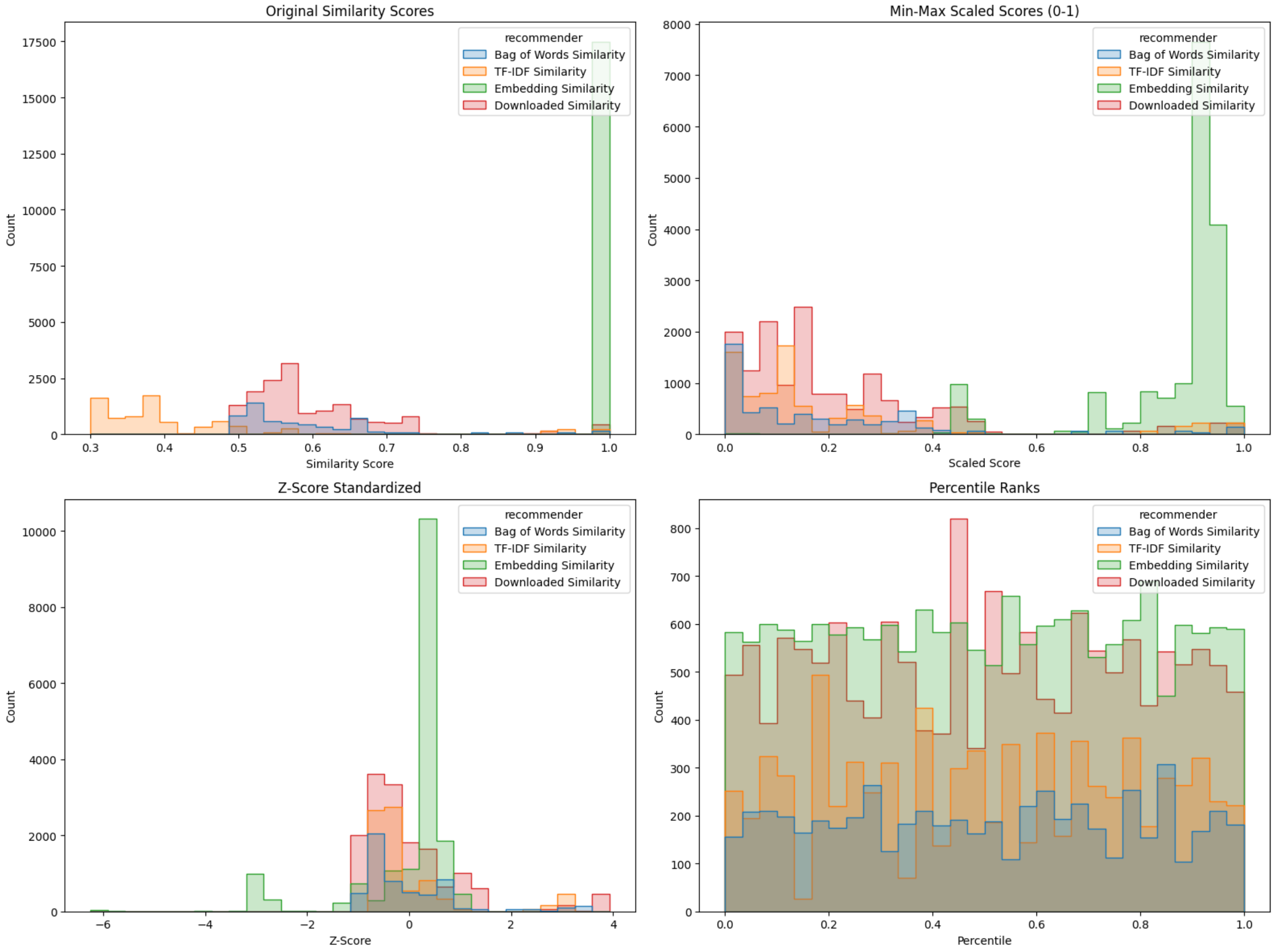




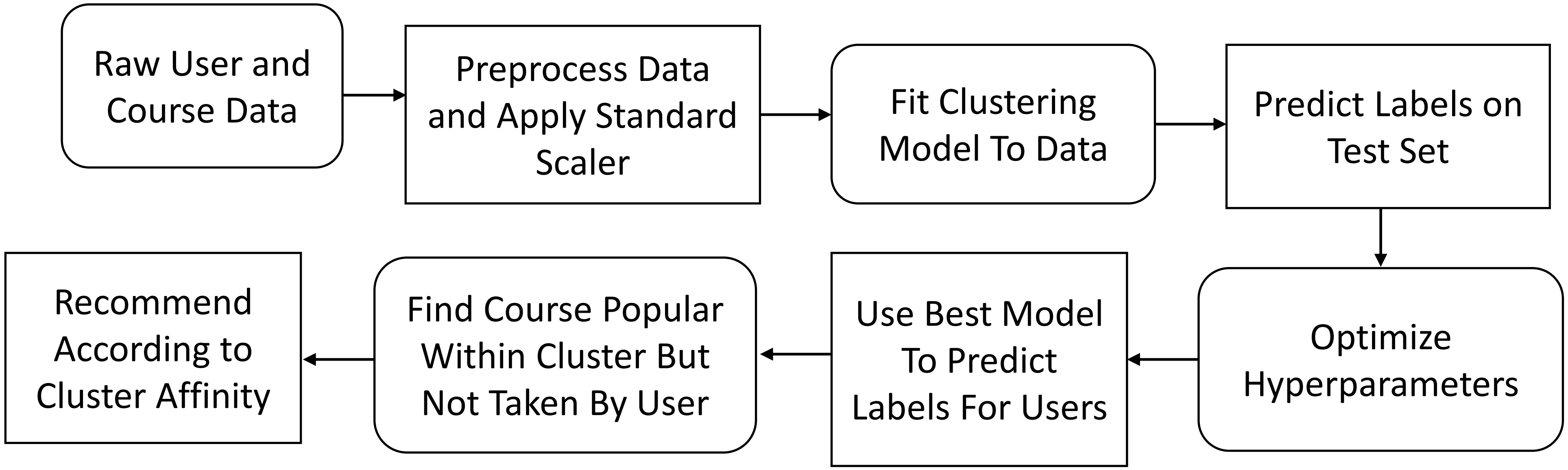


Evaluation results of course similarity based recommender system





Flowchart of clustering-based recommender system



Raw User and Course Data

Preprocess Data and Apply Standard Scaler

Fit Clustering Model To Data

Predict Labels on Test Set

Optimize Hyperparameters

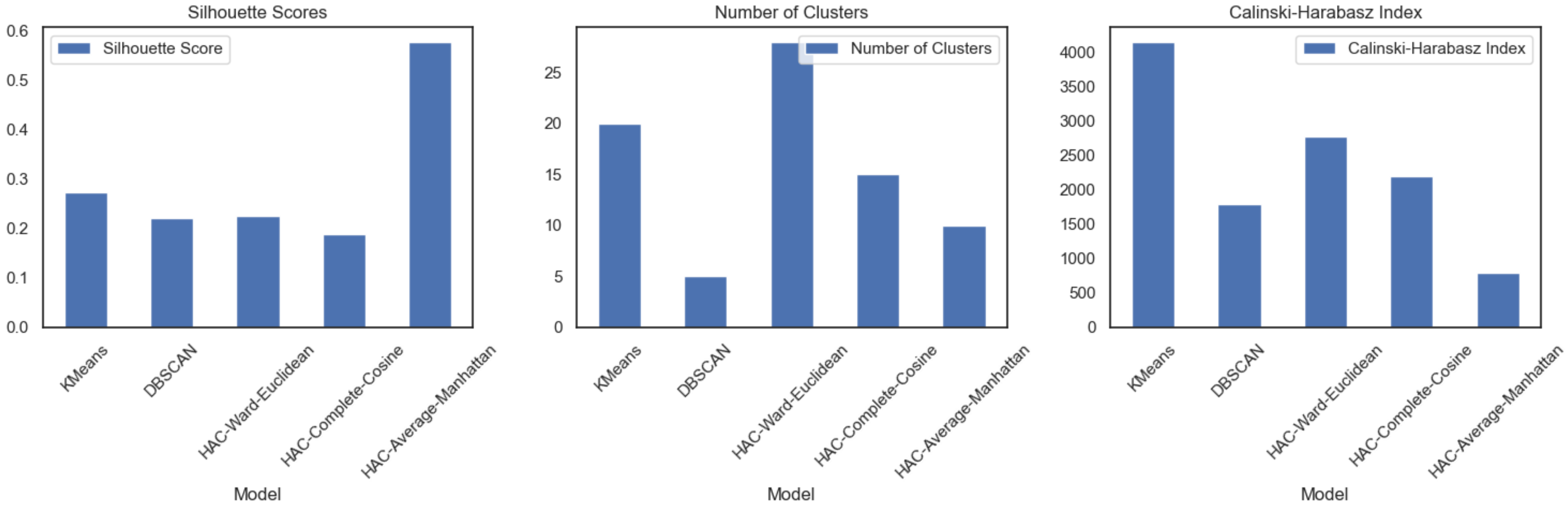
Use Best Model To Predict Labels For Users

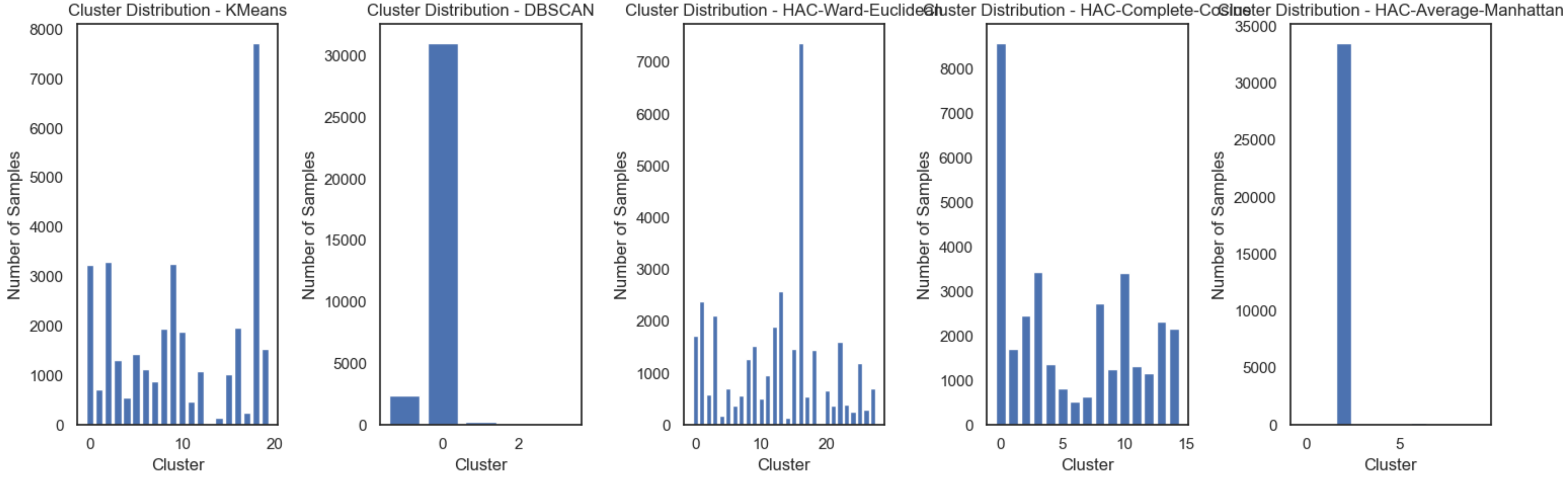
Find Course Popular Within Cluster But Not Taken By User

Recommend According to Cluster Affinity

Evaluation results of clustering-based recommender system

Enrollment Threshold: 10





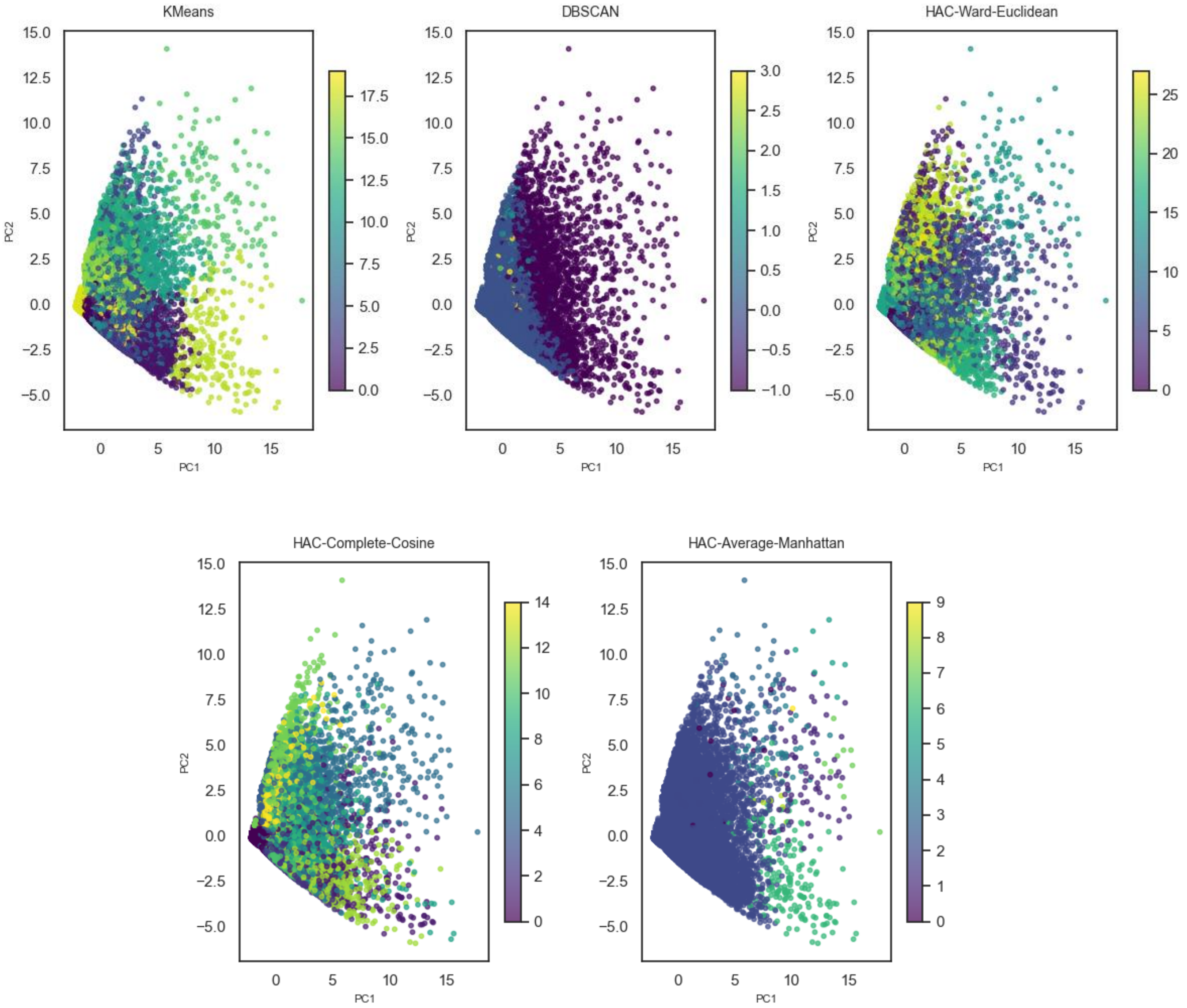
Evaluation results of clustering-based recommender system

Silhouette Score {-1, 1}: Highest Score - 1, Measures how similar an object is to its own cluster

Calinski-Harabasz Index {higher better}: Ratio of between-cluster to within-cluster variance

Davies-Bouldin Index {lower better}: Best Score - 0.0, Average similarity between clusters

Model	Number of Clusters	Silhouette Score	Calinski-Harabasz Index	Davies-Bouldin Index
K-Means	20	0.272050	4159.791551	1.531302
DBSCAN	5	0.221238	1798.678845	1.513762
HAC: Ward-Euc	28	0.225084	2779.051482	1.699835
HAC: Com-Cos	15	0.186803	2206.003037	2.033161
HAC: Ave-Man	10	0.577036	796.655311	1.276104



Evaluation results of clustering-based recommender system

K-Means Clustering:

An okay silhouette score, the highest Calinski-Harabasz Index suggesting good cluster separation, and an okay Davies-Bouldin Index which shows moderate cluster overlap.

DBSCAN:

Lower clustering quality due to lower silhouette score. Lower cluster separation. Similar Davies-Bouldin Index to K-Means, suggesting slight overlap in clusters. From the cluster distributions, it is possible to notice that the clusters are very imbalanced, and most users fit into one cluster. This would not be a great model to use.

Hierarchical Clustering (Ward-Euclidean):

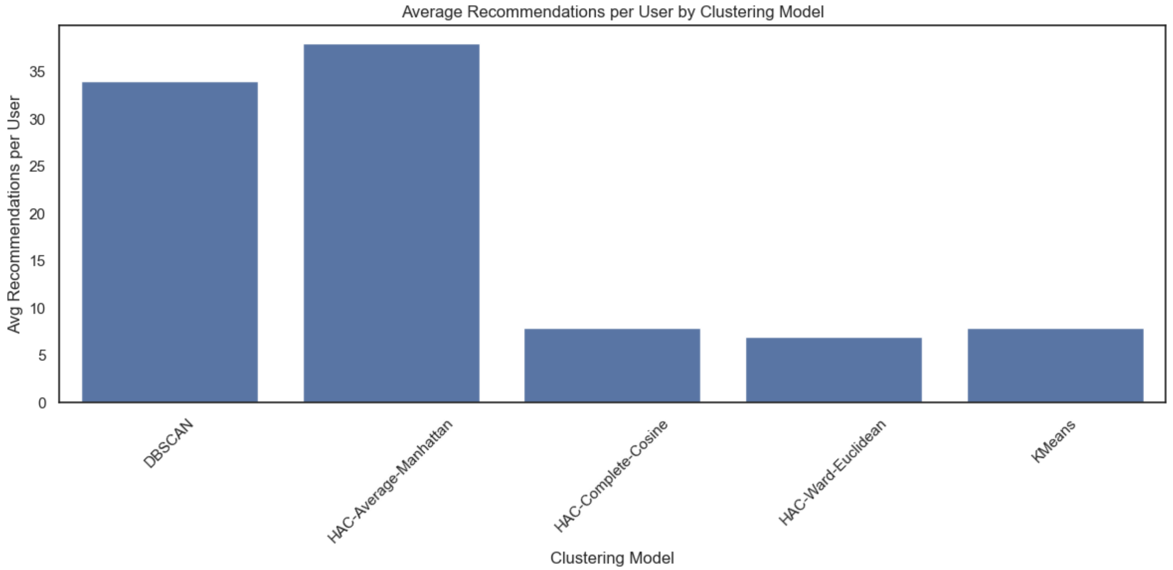
A moderate clustering quality from its silhouette score. It has a good Calinski-Harabasz Index, suggesting good cluster separation. It does, however, have one of the highest cluster overlap, which is not great.

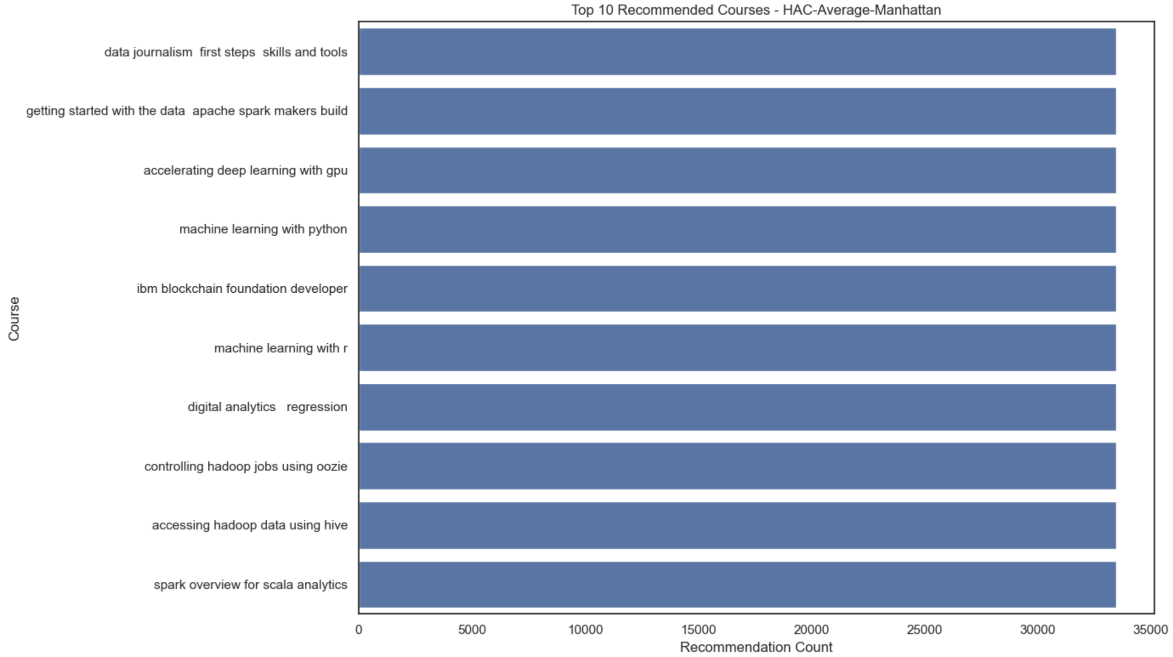
Hierarchical Clustering (Complete-Cosine):

This method, unsurprisingly has the lowest silhouette score (lowest clustering quality), which is expected as cosine is normally used for other tasks and not to separate this kind of data. It has moderate cluster separation and has the highest cluster overlap.

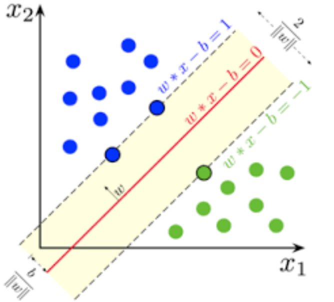
Hierarchical Clustering (Average-Manhattan):

This method has the highest clustering quality and the lowest cluster overlap. However, it has the lowest cluster separation and, similar to DBSCAN, it has one dominant cluster. This would not be a good method for the recommender system.

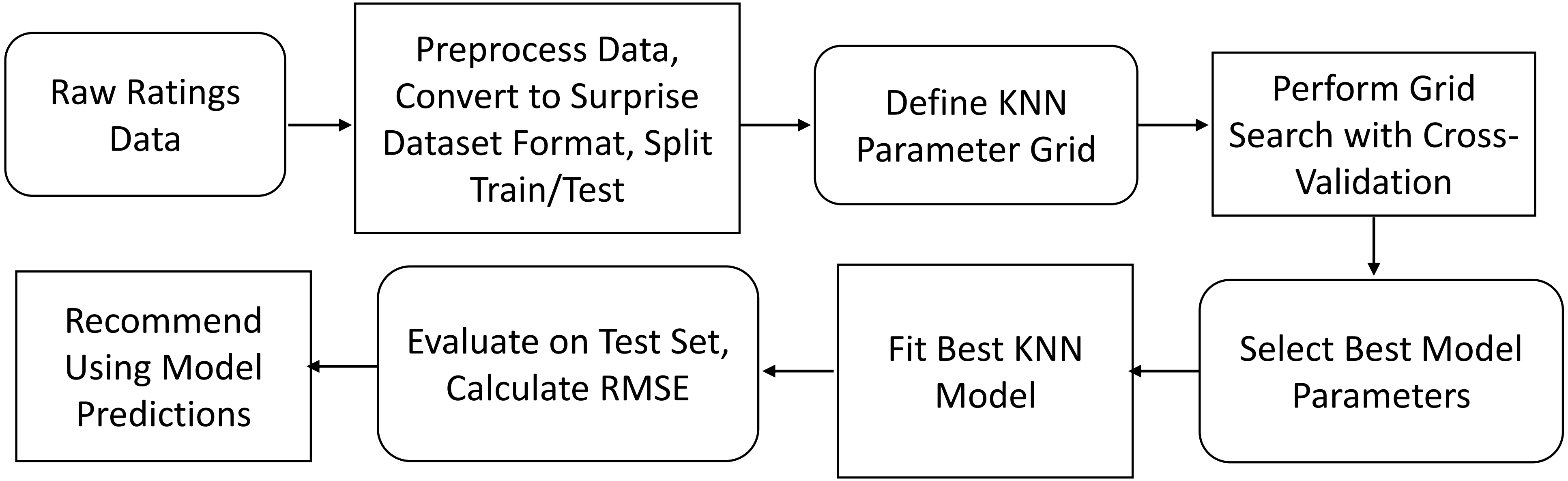




Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based recommender system



Raw Ratings Data

Preprocess Data, Convert to Surprise Dataset Format, Split Train/Test

Define KNN Parameter Grid

Perform Grid Search with Cross-Validation

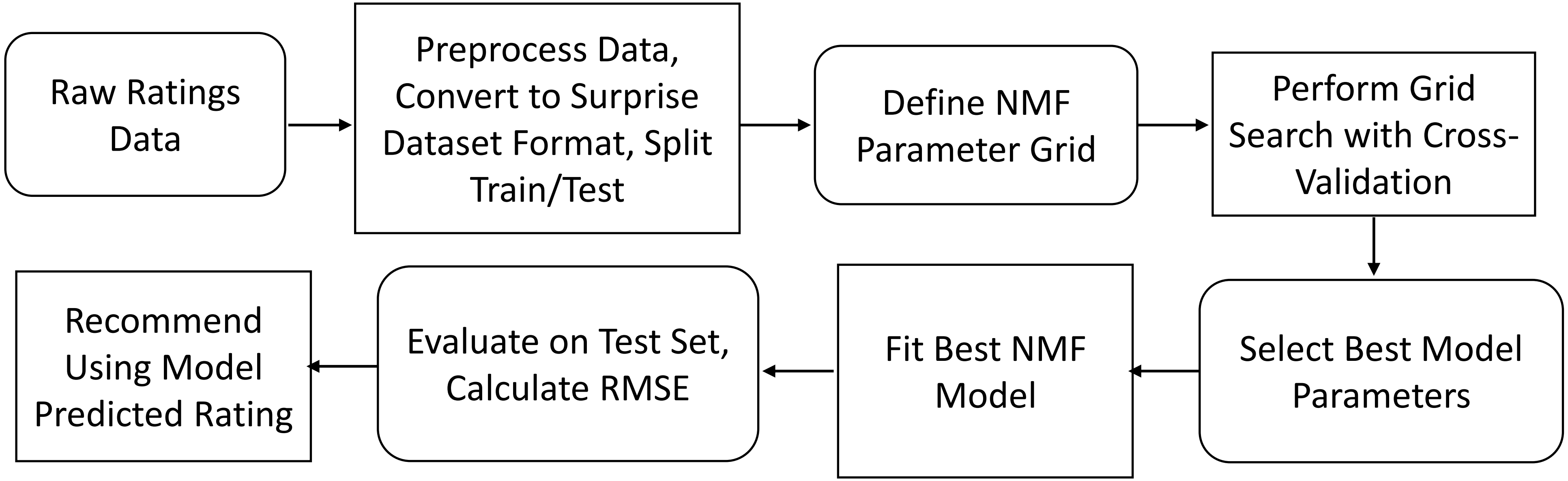
Select Best Model Parameters

Fit Best KNN Model

Evaluate on Test Set, Calculate RMSE

Recommend Using Model Predictions

Flowchart of NMF based recommender system



Raw Ratings Data

Preprocess Data, Convert to Surprise Dataset Format, Split Train/Test

Define NMF Parameter Grid

Perform Grid Search with Cross-Validation

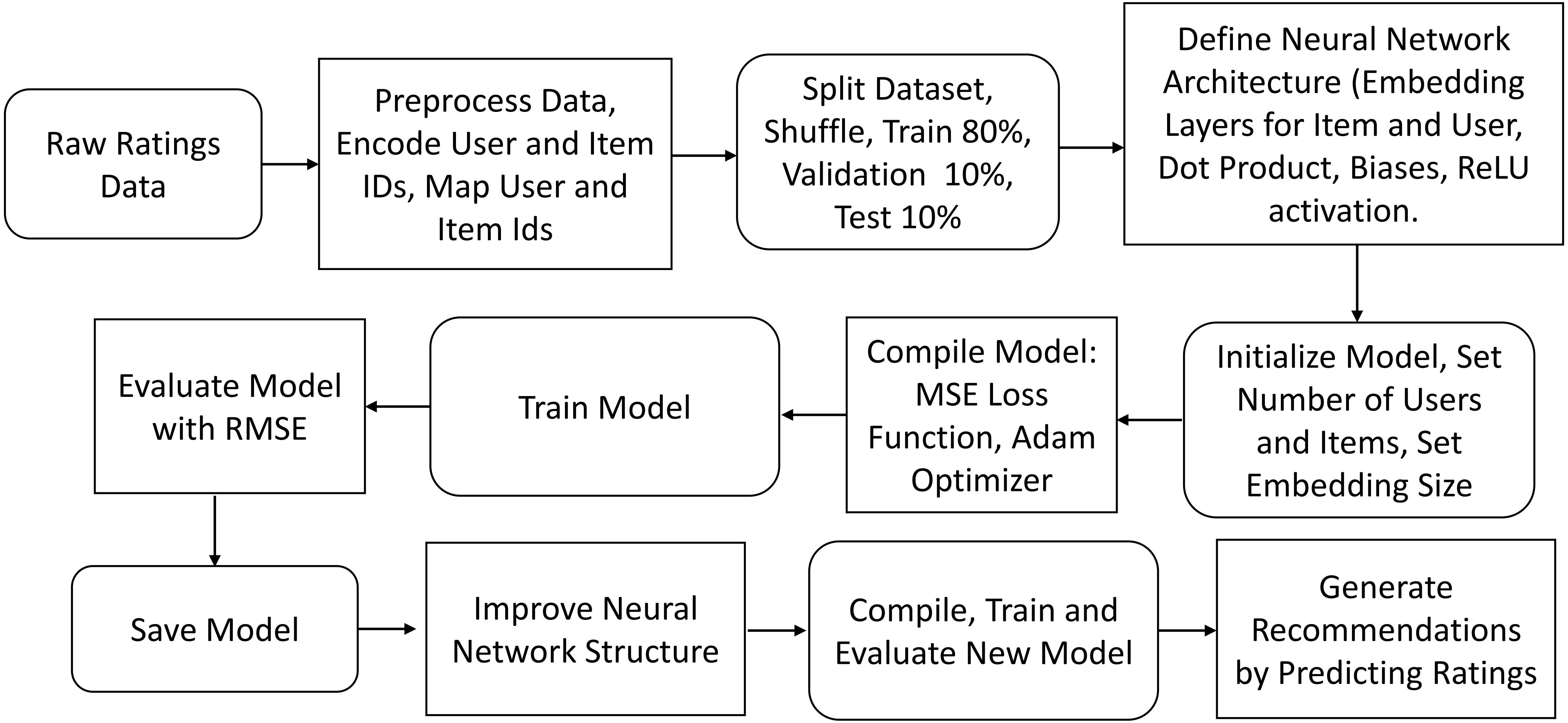
Select Best Model Parameters

Fit Best NMF Model

Evaluate on Test Set, Calculate RMSE

Recommend Using Model Predicted Rating

Flowchart of Neural Network Embedding based recommender system



Raw Ratings Data

Preprocess Data, Encode User and Item IDs, Map User and Item Ids

Split Dataset, Shuffle, Train 80%, Validation 10%, Test 10%

Define Neural Network Architecture (Embedding Layers for Item and User, Dot Product, Biases, ReLU activation.

Initialize Model, Set Number of Users and Items, Set Embedding Size

Compile Model: MSE Loss Function, Adam Optimizer

Train Model

Evaluate Model with RMSE

Save Model

Improve Neural Network Structure

Compile, Train and Evaluate New Model

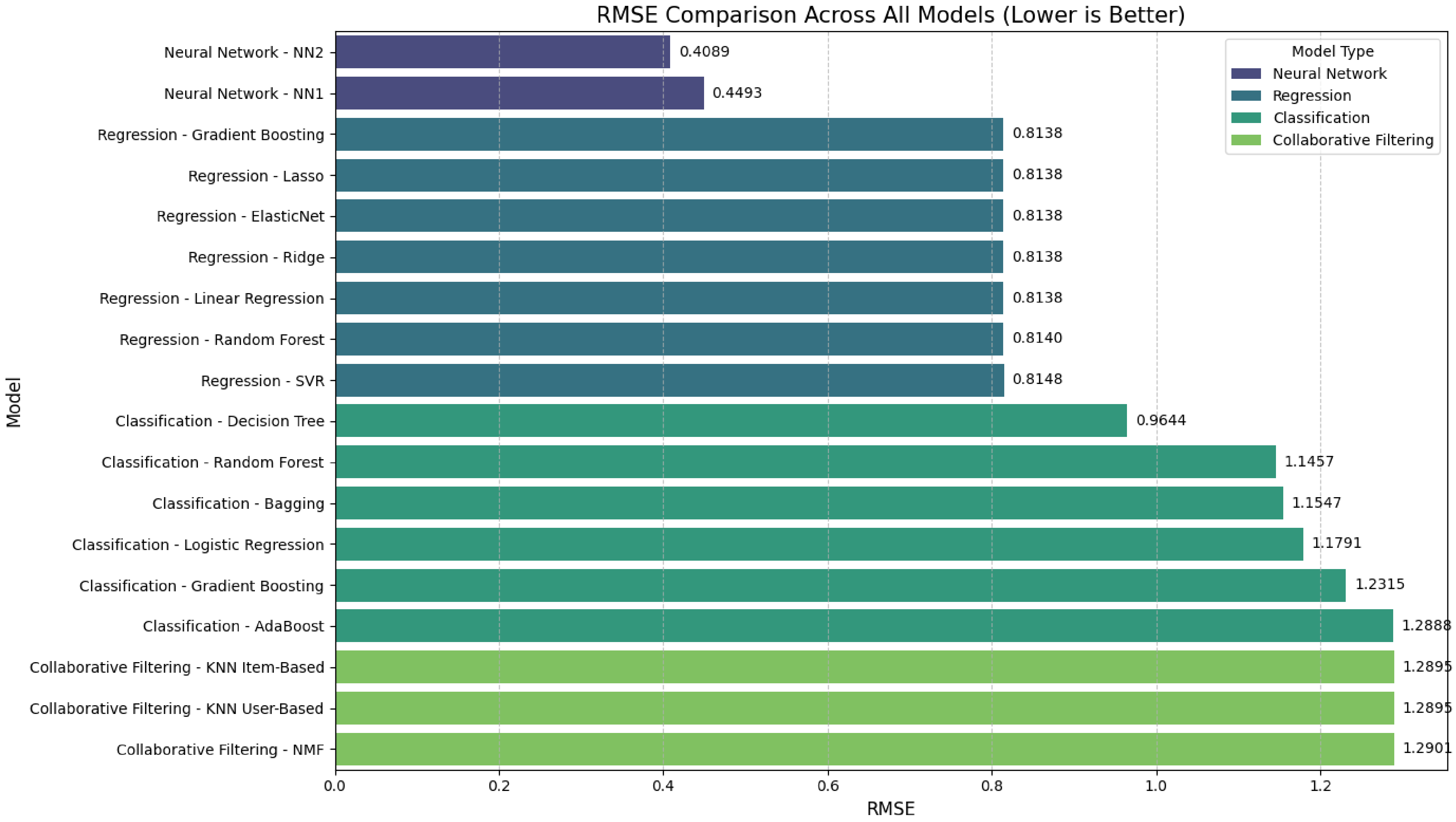
Generate Recommendations by Predicting Ratings

Compare the performance of collaborative-filtering models

Model	RMSE
Neural Network - NN2	0.408922
Neural Network - NN1	0.449327
Regression - Gradient Boosting	0.813813
Regression - Lasso	0.813818
Regression - ElasticNet	0.813818
Regression - Ridge	0.813835
Regression - Linear Regression	0.813835
Regression - Random Forest	0.813990
Regression - SVR	0.814788
Classification - Decision Tree	0.964361
Classification - Random Forest	1.145709
Classification - Bagging	1.154721
Classification - Logistic Regression	1.179098
Classification - Gradient Boosting	1.231527
Classification - AdaBoost	1.288799
Collaborative Filtering - KNN Item-Based	1.289513
Collaborative Filtering - KNN User-Based	1.289513
Collaborative Filtering - NMF	1.290139

Model Type	Mean	Min	Max	Count
Neural Network	0.429125	0.408922	0.449327	2
Regression	0.813985	0.813813	0.814788	7
Classification	1.160703	0.964361	1.288799	6
Collaborative Filtering	1.289722	1.289513	1.290139	3

Compare the performance of collaborative-filtering models



Conclusions

Returning To Hypotheses:

- Collaborative Filtering will perform better than content-based methods.
  - This is true, as collaborative filtering methods are more complex and better suited.
- Classification models will predict the user ratings better than regression models.
  - This was not true at all. The nature of regression models allowed them to predict the ratings much better than the classification models.
- NMF will outperform KNN as methods of Collaborative Filtering.
  - Both methods had very similar results, either one could be used.
- K-Means Clustering will be the best performing unsupervised clustering method for clustering-based recommender systems.
  - Out of the five unsupervised clustering methods, K-Means would probably be the most appropriate one.
- The Neural Network will be the best performing predictor in terms of RMSE.
  - This was true. Both NN, the first and improved versions, both outperform all classification, regression and collaborative filtering methods (from surprise) by a good amount. It makes sense as the NNs are capable of learning more complex relationships in the embedded features.