Course Recommendation System: Architecture, Modifications, and Implementation Report

Recommendation System Project Report

Nov 5 2023

Abstract

This report details the implementation and structural modifications of a Course Recommendation System utilizing the Coursera dataset. The original project design proposed User-to-Item Collaborative Filtering and a dual-tower Deep Learning architecture. However, due to the global nature of the dataset's ratings, the methodology was successfully adapted to employ Content-Based Filtering, Item-to-Item Collaborative Filtering, and an Autoencoder architecture to generate robust, accurate course recommendations based on intrinsic features.

1 Introduction

Recommendation systems are critical for modern digital platforms to surface relevant content to users in an age of information overload. This project aimed to build a comprehensive engine to recommend online courses. The system initially intended to use user interaction history; however, upon inspection of the provided dataset (`Coursera.csv`), it was determined that the data consists of individual courses and global aggregate ratings, rather than individual user-item interactions.

This required a significant methodological pivot to ensure the system remained mathematically sound and practically functional.

2 Dataset Analysis

The `Coursera.csv` dataset contains 3,522 courses with 7 features:

- **Course Name**: Title of the course.
- **University**: The institution offering the course.
- **Difficulty Level**: Beginner, Intermediate, Advanced, or Mixed.
- **Course Rating**: Global average rating (e.g., 4.8).
- **Course Description**: A detailed text description of topics covered.
- **Skills**: Target competencies (e.g., Data Analysis, Python).

Crucially, the absence of a `User_ID` column necessitated the shift away from traditional User-to-Item Matrix Factorization (SVD).

3 Methodology & Implementation

3.1 1. Content-Based Filtering (NLP Approach)

The fundamental premise of Content-Based Filtering is recommending items similar to those a user has liked in the past, based on item attributes.

3.1.1 Algorithm Details

- **Feature Engineering:** The ‘Course Name‘, ‘Difficulty Level‘, ‘Course Description‘, and ‘Skills‘ columns were concatenated into a single master ‘tags‘ string for each course.
- **Text Preprocessing:** The strings were converted to lowercase, stripped of punctuation, and processed using the Natural Language Toolkit (NLTK) ‘PorterStemmer‘ to reduce words to their root forms (e.g., ”programming” to ”program”).
- **Vectorization:** A ‘CountVectorizer‘ was utilized to convert the text documents into a matrix of token counts, limited to the top 5,000 most frequent words, excluding English stop words.
- **Similarity Metric:** Cosine Similarity was computed across the vectors to find the mathematical angle between course descriptions. Courses with the smallest angles (highest cosine similarity scores) were recommended.

3.2 2. Item-to-Item Collaborative Filtering

To replace the non-functional SVD approach, an Item-to-Item filtering mechanism was constructed. This approach identifies ”similar” courses based on their quantitative metadata rather than user behavior.

3.2.1 Algorithm Details

- **Feature Selection:** The ‘Course Rating‘ (continuous), ‘Difficulty Level‘ (categorical), and ‘University‘ (categorical) were selected.
- **Encoding & Scaling:** Categorical variables were converted to numeric representations using ‘LabelEncoder‘. The feature matrix was then normalized using ‘StandardScaler‘ to ensure that features with larger magnitudes (like high arbitrary label codes) did not disproportionately influence the distance calculations.
- **Similarity Metric:** Cosine similarity was applied to the scaled feature matrix to generate recommendations. For instance, a highly-rated Beginner machine learning course from Stanford would recommend other highly-rated, introductory statistical/computation courses from elite universities.

3.3 3. Deep Learning (Autoencoder Architecture)

Deep Learning allows for the discovery of complex, non-linear relationships between variables that simple distance metrics might miss. The initial project design contained a classic Collaborative Filtering Neural Network (requiring User IDs). This was replaced with an Unsupervised Autoencoder.

3.3.1 Architecture

An Autoencoder is a type of artificial neural network used to learn efficient codings of unlabeled data.

- **Encoder:** The input layer receives the normalized course feature numerical vector. This is passed through a Dense layer with a `ReLU` activation function, compressing the multi-dimensional features down into a 2-Dimensional embedding space.
- **Decoder:** The 2D embedding is expanded back out to the original dimensions via a linear Dense layer.
- **Training:** The model was compiled using the Adam optimizer (learning rate = 0.01) and trained to minimize the Mean Squared Error (MSE) between the original input features and the reconstructed output for 50 epochs.
- **Extraction:** Once trained, the Decoder is discarded. The Encoder is used to predict the 2D embeddings for all courses. Cosine similarity is then performed on these low-dimensional embeddings to generate final recommendations.

4 Modifications and Fixes Summary

1. **Environment Isolation:** Resolved `ModuleNotFoundError` issues by creating a clean virtual environment (`venv`) and generating a strict `requirements.txt` dependency file.
2. **Interactive CLI:** Developed `clean_recommender.py`, a robust, infinite-loop terminal script that allows users to query courses interactively with built-in fuzzy matching and error handling.
3. **Jupyter Notebook (`RS Project.ipynb`):** Completely rewritten to execute flawlessly end-to-end, replacing the broken SVD and Keras sections with the mathematical robust Item-to-Item and Autoencoder models described above.
4. **Workspace Reorganization:** Archived obsolete PDF literature and broken code fragments into an '/Archive' directory, resulting in a production-ready repository.

5 Conclusion

The Course Recommendation System has been successfully rescued from architectural flaws caused by dataset limitations. By pivoting to Content-Based, Item-to-Item, and unsupervised Deep Learning methodologies, the system now provides highly accurate, multi-faceted recommendations for the Coursera dataset while maintaining a pristine, reproducible codebase.