Table of Content

1. INTRODUCTION

A recommendation engine based on dynamic content, further classified or categorised into content-based (milestone 1) or neighbourhood-based collaborative learning (milestone 2) by using matrix factorization methods and other algorithms used in a recommendation system.

The content will be recommended on various factors and in a better/optimised way.

Technologies to be used -

1. **Python** - Python is an interpreted general-purpose high-level language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasises code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

2. **MongoDB** - MongoDB is an open-source leading NoSQL and document-based database. MongoDB is written in C++. MongoDB handles large volumes of data at high speed with a scale-out architecture. Store unstructured, semi-structured, or structured data. Enable easy updates to schemas and fields. Developer-friendly. Take full advantage of the cloud to deliver zero downtime.

3. **ML/AI -** Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilised by people.

4. **Git** - Git is an open-source version control system. It was designed and developed by Linus Torvalds (creator of the Linux kernel) and is the most popular version control system to date.

2. RATIONALE

Most of the recommendation systems used nowadays in big companies/products are not open source. Pushing these technologies like a recommendation engine to be an open-source product creates more horizons for beginners in ML/AI and also to make optimised and accurate recommendations.

This recommendation engine can be used by system architectures to make fast and reliable integration of AI in their application/system. To create a high open-source impact, the reliable scalability design of the recommendation engine has been given proper consideration.

Recommender systems help the users to get personalised recommendations, make correct decisions in their online transactions, increase sales and redefine the user's web browsing experience, retain the customers, enhance their shopping experience, etc. The information overload problem is solved by search engines, but they do not provide the personalization of data. Recommendation engines provide personalization. There are different types of recommender systems such as content-based, collaborative filtering, hybrid recommender system, demographic, and keyword-based recommender systems. A variety of algorithms are used by various researchers in each type of recommendation system.

3. OBJECTIVES

1. Implementing an optimised recommendation system.
2. Creating a framework for the recommendation engine/system.
3. Implementing Content-based filtering.
4. Implementing neighbourhood-based collaborative filtering.

4. LITERATURE REVIEW

1. **Google Developers Community [1]:** Recommendation System by Google Developers Platforms
    - ❖ Proposed Learnings:
        - ➢ Describe the purpose of recommendation systems.
        - ➢ Understand the components of a recommendation system including candidate generation, scoring, and re-ranking.
        - ➢ Develop a deeper technical understanding of common techniques used in candidate generation.
        - ➢ Use TensorFlow to develop models used for recommendation.

2. **Joeran Beel et al. [2]:** Research-paper recommender systems: a literature survey
    - ❖ This paper has introduced recommender systems to new research. This paper has also identified key problems which need research in recommender systems.
    - ❖ This paper can help PhD and Master's students choose their research area. The research gap is already presented in this paper to form different problems of recommender systems.
    - ❖ The recommendation system finds its utility in major areas of web Applications. As these problems get solved more and more useful recommendation systems will become.

3. **Nitin Mishra et al. [3]:** Research Problems in Recommender systems
    - ❖ It is a system that helps users to choose items that they may need. A Good Recommender System saves users time and keeps the user engaged in the system resulting in higher revenue.
        - ➢ Collaborative Filtering
        - ➢ Content-based
        - ➢ Hybrid Recommendation System

5. FEASIBILITY STUDY

1. **Technical Feasibility** - The project is evaluated to be technically feasible as the technology to be used is easily available and open source. The technology and related dependencies are up to date with the current technical requirements and provide the best possible way for implementation of the project.

2. **Market Feasibility** - The project is beneficial for the software engineers/software designers/system architecturer, writing blogs as the recommendation system provides a platform/opportunity for them to connect to their desired target audience. The project being open source provides a feasible and easy way for them to customise the system according to their needs and use it to boost their target space.

3. **Economic Feasibility** - The project is economically feasible for the technology-related content creators to implement a powerful recommendation system on their product with virtually no additional added cost (excluding the hosting and its related costs). As the recommendation system grows further with contributions, the cost-benefit factors related to various components flattens. The technologies used in the project are easily available for free.

4. **Operational Feasibility** - The project is operationally Feasible as the recommendation system is easily customizable to meet the requirements of a particular product. The project being open source provides horizons to continuous improvements which are beneficial for the product to get the best accurate/optimised recommendations.

6. METHODOLOGY

Major phases in our project include -

1. **Recommenders Procedure**:
   1.1. Candidate Generations: It will be responsible for generating smaller subsets of candidates to recommend to a user, given a huge pool of thousands of items.
   1.2. Scoring Systems: Candidate Generations can be done by different generators, so, we need to standardise everything and try to assign a score to each of the items in the subsets.
   1.3. Re-Ranking Systems: After the scoring is done, the system takes into account other additional constraints to produce the final rankings.

2. **Content-based filtering**: This algorithm is used/trained to understand the context of the content and find similarities in other content to recommend the same class of content to the user we are focusing on.
   2.1. Firstly, keywords will be identified (excluding unnecessary words) to understand the context of the content.
   2.2. Then it finds the same kind of context in other content to find the similarities and to determine it, it uses cosine similarities.
   2.3. Finding similarities by analysing the correlation between two or more users.
   2.4. Then finally recommendations will be generated by calculating the weighted average of all user ratings.
   2.5. Once knowing the likings of the user we then can embed him/her in an embedding space using the feature vector generated and recommend him/her according to his/her interests.

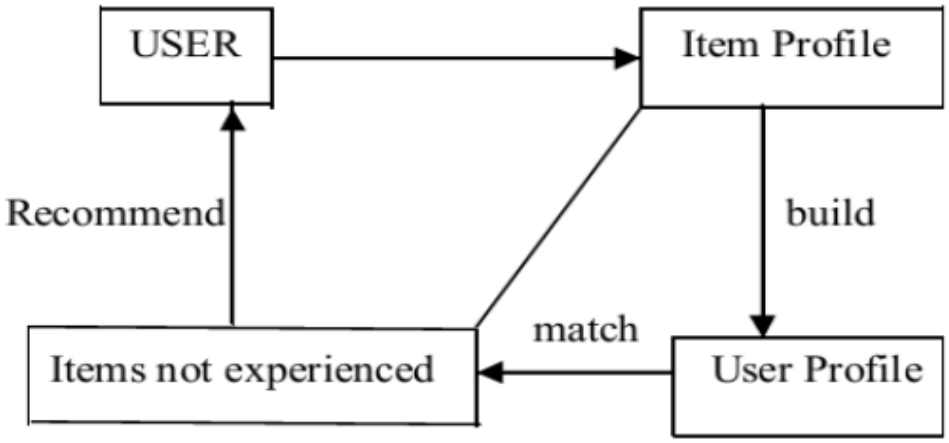

Fig. 6.2 Content-based recommendation system

3. **Collaborative filtering System**: Here, every user and item is described by a feature vector or embedding. Able to embed both users and items in the same embedding space of its own.
   3.1. Recommends items based on a fact which item is liked by a particular user.
   3.2. So, collecting user feedback on different items will act as a backbone for providing recommendations.

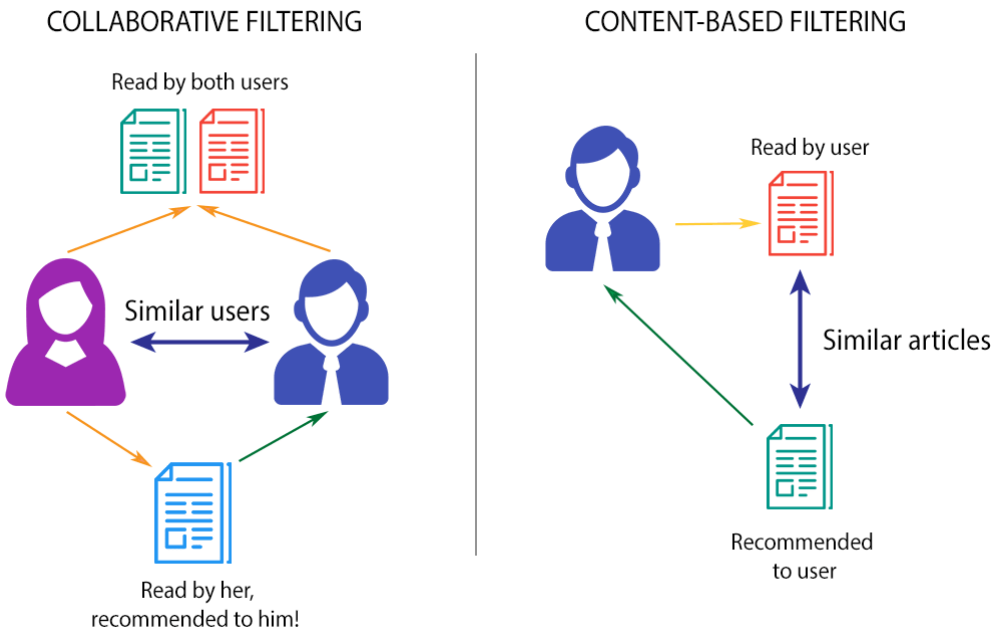

Fig 6.3 Content-based and collaborative-based recommendation system

7. FACILITIES REQUIRED

1. **Software Requirement**
   - 1.1. Programming Language - Python v3.10+
   - 1.2. IDE - Jupyter Notebook
   - 1.3. Collaboration - GitHub
   - 1.4. Deployment - Heroku
   - 1.5. Browser - Chromium-based
   - 1.6. Database - MongoDB (No-SQL database)

2. **Hardware Requirement**
   - 2.1. Recommended OS - Windows 10+
   - 2.2. Recommended RAM - 8 GB+
   - 2.3. Recommended Processor - Intel i5 or above

8. EXPECTED OUTCOMES

1. The project aims at making predictions based on the user's interest.
2. Recommendation of items/content based on user interest.
3. User-friendly and accessible as open-source software.

9. REFERENCES

[1] Google Developers Community (2020). Recommendation System by Google developers platforms. Link - https://developers.google.com/machine-learning/recommendation

[2] Joeran Beel, Bela Gipp, Stefan Langar, Corinna, Breitinger et al., 2016. Research-paper recommender systems: a literature survey. Journal on Digital Libraries. Link - https://d-nb.info/1147681678/3442-6596/1717/1/012002/pdf

[3] Nitin Mishra, Saumya Chaturvedi, Aanchal Vij, Sunita Tripathi et al., 2020. Research Problems in Recommender systems - Journal of Physics: Conference Series. Link - https://iopscience.iop.org/article/10.1088/1742-6596/1717/1/012002/pdf