he recommendations based on the commonality between users preference/ratings and the preferences/ratings of other users.

Content-Based Recommender Systems that provide recommendations to the user by automatically their preference to the contents of the products. Like recommendations for the movies, music and web pages. In content-based systems, the items are described by common attributes. User preference is predicted by analyzing the relationship between the product attributes and ratings by the user.

**TODO:**

**Begin planning what the collaborative filtering model could be, based on the data model of the open data set. Documentation here:**

[**https://communitydata.science/scratch-data/**](https://communitydata.science/scratch-data/)**Introduction**

The aim of this project is to find out ways to recommend projects and interest to users of Scratch, a visual programming language platform by MIT. Scratch is also a community-based platform that encourages sharing and mixing of code with other users. Scratch users can develop their own code or build upon the work of others.

Currently, there is no way to find the projects that are of interest to a specific user. Also, there is a very little personalised recommendation of projects to users. In the present scenario, the users are shown different featured projects either handpicked by the community moderators or what is being mostly mixed and liked by other users.

**Background**

Scratch is one such TEL platform developed by Lifelong Kindergarten Group at the MIT Media Labs. Scratch helps young and inexperienced students to develop programming skills and think creatively. With Scratch, one can program interactive stories, games, and animation. It also provides a collaborative platform through which students can share their own code and develop on others work. Scratch is also being used in more formal education by teachers to introduce programming concepts to inexperienced users. It is often been observed that some users get demotivated easily because of either they are unsure of where to go further or the programming exercises are not up to their individual expectations. Thus, the concern arises how do we keep students motivated to Scratch and improve the learning experience.

An effective solution is to recommend students with projects from other users according to their level of knowledge and previous experience. This intuitively is known as a Recommender Systems (RSs), a system that recommends users with exercise based upon their previous activities. Recommender system in an educational environment is proven to be significantly beneficial.

**Aim**

In this research, we try to explore ways to improve the user experience of the scratch community by recommending tailored content to the users. A follow up of this would be to study the effect of recommender system on users interaction pattern.

**Research Question**

With the above aim in mind, the research question for this dissertation becomes

*“Is it possible to provide personalized and tailored recommendations in Scratch for projects and other components.”*

**Research Objective**

In order to achieve the stated research question, the following objectives must be fulfilled:

1. A recommendation engine backend that generates a recommendation based on the user's profile.
2. A browser extension to show users the recommendation on the front page of the Scratch platform.

**State of the Art**

This topic will discuss the current state of the art methods for content personalization with the help of the recommender system and user data. We will also discuss the current practices in recommender systems in TEL platform.

**Recommender system:**

**What is a recommender system?**

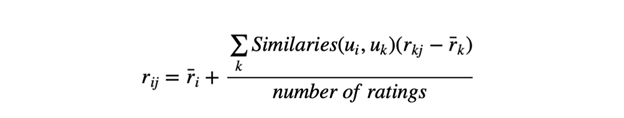
A recommender system is an information filtering system that tries to predict the choices to the user is likely to like.

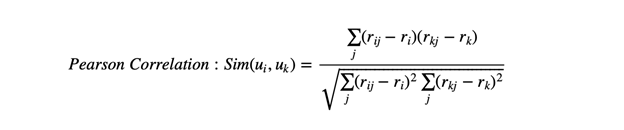
In general existing recommender system can be generally be classified into two main types

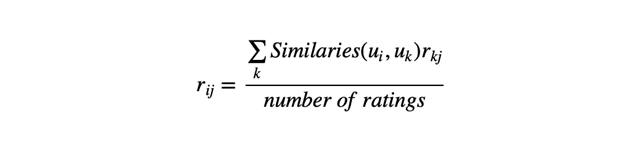
Collaborative Filtering Recommender Systems estimates t

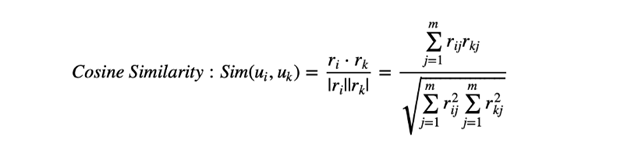
**Nearest Neighborhood**

The standard method of Collaborative Filtering is known as Nearest Neighborhood algorithm. There are user-based CF and item-based CF. Let’s first look at User-based CF. We have an n × m matrix of ratings, with user uᵢ, i = 1, ...n and item pⱼ, j=1, …m. Now we want to predict the rating rᵢⱼ if target user i did not watch/rate an item j. The process is to calculate the similarities between target user i and all other users, select the top X similar users, and take the weighted average of ratings from these X users with similarities as weights.









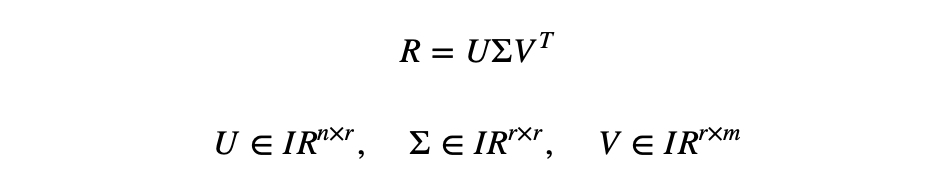
Basically, the idea is to find the most similar users to your target user (nearest neighbours) and weight their ratings of an item as the prediction of the rating of this item for the target user.

Without knowing anything about items and users themselves, we think two users are similar when they give the same item similar ratings. Analogously, for Item-based CF, we say two items are similar when they received similar ratings from the same user. Then, we will make a prediction for a target user on an item by calculating a weighted average of ratings on most X similar items from this user. One key advantage of Item-based CF is the stability which is that the ratings on a given item will not change significantly over time, unlike the tastes of human beings.

**Matrix Factorization**

For solving sparsity and scalability problem with standard CF method

To see how a matrix being factorized, the first thing to understand is Singular Value Decomposition(SVD). Based on Linear Algebra, any real matrix R can be decomposed into 3 matrices U, Σ, and V. Continuing using movie example, U is an n × r user-latent feature matrix, V is an m × r movie-latent feature matrix. Σ is an r × r diagonal matrix containing the singular values of the original matrix, simply representing how important a specific feature is to predict user preference



To sort the values of Σ by decreasing absolute value and truncate matrix Σ to first k dimensions( k singular values), we can reconstruct the matrix as matrix A. The selection of k should make sure that A is able to capture the most of variance within the original matrix R, so that A is the approximation of R, A ≈ R. The difference between A and R is the error that is expected to be minimized. This is exactly the thought of Principle Component Analysis.

How do we find optimal qᵢ and pᵤ? Like most of machine learning task, a loss function is defined to minimize the cost of errors.

