**Influence Based Recommendation Model**

The Influence Based Recommendation Model aims at generating a set of suggested courses for the user at the time of checkout based on the course they are currently buying. This can be used as the **frequently bought** solution to the course.

The Code is elaborated in an extensive way as follows:

**Input**

The code takes input in the form of a json file(input\_icr.json). This file must contain every feature the user wants to put in, in a detailed format:-

{

"\_id":{"$oid":"5ecea1b690cfa44a44f83139"},

"user\_id":"Caramel\_1",

"courseId": "1",

"course\_name":"Learn and Understand NodeJS",

"user\_feedback":"4.2",

"user\_score":"85"

}

Since all the user data has been already gathered in previous iterations and at the start of the courses they completed, there is no need to get access to more data. This is done to reduce redundancy.

This program was initially made in IPython compiler and for converting to a .py file a lot of dependencies have to be installed.

Since this is an autonomous Process and no user input is required. The data from this model will be showed as suggestions after the user has completed a course

**Code**

* Data is taken from the json input of CourseName, Current course, user feedback, user score and current sale discount.
* Every course is firstly indexed inside the dataset. In practical use this has to be done only once.
* All features are saved as tags into a list and empty data slots are filled and handled. Similarily , same operation is performed over user score and rating. These tags list are then added to the initial dataframe for better referencing.
* A CountVectorizer() object is created. This converts a collection of text documents to a matrix of token counts. Where in the tokens are the “tags”, “Ratings”, “UserScore”. The count matrix created by this operation is feeded into cosine similarity function. Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space. So it will measure the similarity between the vectors found by the tags in the matrix.
* Two functions are created to call the course names and to find their indices - get\_Course\_Name\_from\_index(index) and get\_index\_from\_Course\_Name(Course\_Name)
* Finally, a list is enumerated with the help of the above two functions. This list is sorted based on the rating of the courses. From this list, elements are called in a loop and saved onto a dataframe. And this dataframe is converted into json format.

\*\*Impt to note that the database has to be sorted by the “User Review” column before operation\*\*

**Output**

The final dataframe is then converted to json format and saved into a separate file with the name as output\_irc.json

Example output

{

"influenced\_courses":

{

"0":"Build NodeJS applications with Mongodb",

"1":"Complete Modern Amazon clone: Angular 5 and Node.js",

"2":"Complete Python Bootcamp: Go from zero to hero in Python 3",

"3":"Node.js: The Complete Guide to Build RESTful APIs (2018)",

"4":"Deep Learning Prerequisites: Linear Regression in Python"

}

}