Software Documentation – CS 410, Fall 2021

Team Name – RS4

Team Members –

1. Rudranil Deb Roy – rdebroy2
2. Shaleen Mehrotra – shaleen3 **(Captain)**
3. Shruti Kalia – skalia2
4. Sujan Das – sujan2
5. Sreyashi Das – das25

Project Topic – Free Topic **– IntelliReco (An intelligent recommendation system)**

1. **Overview of the function of the code**
   The main functionality of the application is to provide the users with recommendations of products from online marketplace(s) based on their keyword input criteria. The system chooses products from an SQLite database populated offline (through python files in this same application). The application provides a User Interface to interact with the system and get recommendations by supplying a few product related keywords. These keywords could be the specifications of the product that the user is interested in. Once everything is entered, user clicks the recommend button and a list of recommended products is displayed on the screen. These returned products are in accordance with the category selected by the user and the keywords entered by him/her. The UI also presents the URL of the specific product which will directly take user to the product page at the marketplace. The system searches the downloaded description of some specific products from amazon marketplace which is stored in our SQLite database tables. The UI also provides a dashboard for an easy graphical overview of the products covered in the system. The graphs presented are - **Categories Overview** which is a pie chart of different categories with corresponding product counts/percentage and **Euclidean Distance** between the texts of the products corresponding to a specific category.

   Note: Though the system is operating based on an SQLite database (offline), the system has got the capability to download as many products as we choose, and the UI dynamically provides recommendation based on all the downloaded information.

2. **How the software is implemented**
   The code is divided into 2 parts.
   First part is the backend code that is written in Python. This backend code has multiple parts to it –
   a. Python file **searchresults.py** – The code in this file is used to crawl over the amazon web pages and read about all the products available on that page. The URLs of all the pages that we want this code to crawl over is given in **search_results_urls.txt** file. The details that are read after crawling over the web pages are stored in the database. The name of the database is **intellireco2.db**.
   b. Python file **DatabaseProvider.py** – This file contains the functions that create a SQL database and the functions that are used to connect to that database.

c. Python file **CosineSimCalculator.py** – This file contains the functions which take the keywords that were entered by the user and calculates the cosine similarity between those keywords and the description of the products that are stored in the database. The description of the products go through stemming, tokenization and stopwords are removed from it. This file uses natural language toolkit to do the tokenization and stemming of the product description.
d. Python file **server.py** – This file contains the API methods. When we click "Recommend" button on the UI, the **searchResultQueryArr** method in this file is called which reads the category from the database. Then the list that is read from the database is ordered according to their similarity to the text that was entered. The ordered list is then returned to the UI component.
In the method **getEuclideanProductDistance**, a different approach is conveyed through Euclidean distance by comparing product text as vectors. In this process we have used python sklearn package. On the dashboard, these Euclidean distances are shown as a scattered plot graph developed using react-google-charts.

Second part is the UI that is written using React and JavaScript. There are multiple UI components –

We have used a React material admin template which has various UI files that support the whole user interface. We have updated various components to fir our designs. Some of the important ones are mentioned below.

a. JavaScript file **Tables.js** – This file contains the structure of the table where the result of product recommendation is rendered and shown on the UI page.
b. JavaScript file **Login.js** – This file contains the login page of our application.
c. JavaScript file **Dashboard.js** – This file contains the setup for pie chart and scatter plot.

Note: We have customized on top of Amazon Scraper using Selectorlib (https://github.com/scrapehero-code/amazon-scraper.git) to read Amazon Marketplace product details. Also, we have developed on top of React Material Admin UI template - https://github.com/flatlogic/react-material-admin.git

3. **How to install and run the software**

The steps to follow to run this software are –

a. Clone the git repository and navigate to Intelligent-Recommendation\flask-server.
b. pip install -r requirements.txt.
c. Navigate to dataload folder.
d. python server.py command will launch the backend webserver at localhost:5000 port.
e. Navigate to Intelligent-Recommendation\intel-reco-ui.
f. npm install. This command will download all the required react libraries needed by the project.
g. npm start. This command will start the UI at localhost:3000 port.
h. Click on login on the first page in the UI. Then click on Recommendation tab on the left side.
i. Select the category of the product you want to search and enter some keywords for that product.
j. Click on Recommend to view the results.

Optional steps to update the database –

a. Add the amazon page link of any item that you want to load in the database (intellireco2.db) to the **search_results_urls.txt** file. For e.g., https://www.amazon.com/s?k=goggles.
b. Run the python file **searchresults.py** to load the data from this new link to the database.

4. **Contribution of team members**
    a. **Rudranil Deb Roy** – Setting up the environment and initial project ready. React UI development. Text similarity approaches and implementation. Dashboard graphical charts. Research of the initial marketplace product data. Integration testing and evaluation. Working on the connectivity between React UI and python backend.
    b. **Shaleen Mehrotra** – React UI development. Text similarity approaches and implementation. Research of the initial marketplace product data. Integration testing and evaluation. Reading product details, web crawling and populating SQLite database.
    c. **Shruti Kalia** – React UI development. Text similarity approaches and implementation. Integration and testing. Research of the initial marketplace product data. Integration testing and evaluation. Reading product details, web crawling and populating SQLite database.
    d. **Sujan Das** – React UI development. Dashboard graphical charts. Text similarity approaches and implementation. Research of the initial marketplace product data. Integration testing and evaluation. Working on the connectivity between React UI and python backend.
    e. **Sreyashi Das** – React UI development. Text similarity approaches and implementation. Integration and testing. Research of the initial marketplace product data. Integration testing and evaluation. Working on the connectivity between React UI and python backend.