Product Recommendation System

A report submitted for the internship under

AI Smart Bridge

Submitted By

Utkarsh Tikkiwal (20BCE0334)
Pranav Karande (20BCE0450)
Tejas Sharad Mahajan (20BCE2219)
Ujjwal Akash (20BCE2202)

School of Computer Science and Engineering

July 2023

1. INTRODUCTION

1.1 Overview

In the online marketplace world, companies make use of user history and product data to make recommendations to users. The data collected to make these recommendations are search terms, purchase history, product category, items in cart information and many more. Most commonly, recommendation engines make use of collaborative filtering (purchases/views by similar customers in the same category) to identify products that consumers may like. Another alternative and less widely used technique is to make use of images to compare product similarity. Images may be better representative of consumers interest than text. Further, our text and brand-based recommendation is also more advanced as consumers place a high value on brands, and important keywords relating to products frequently occur in the title of products.

In this project, we present a recommendation engine for online shopping that takes an image as an input and then tries to understand the information about the features from the images. We use a convolutional neural network to classify the input image as one of the product categories. Then we use neural network and cosine similarity on the extracted feature mappings to calculate the similarity between product images and product description, which will be used to recommend other similar products from our database.

1.2 Purpose

The aim of the project is to make recommendations based on images, title and brand information and to classify Amazon products. Our project works in real-time, providing recommendations for the user based upon a product of their choice. The user simply enters the product url in a text file, and runs the program, specifying the method of recommendation and the number of Amazon pages to search for similar products.

The first part of the project trains and tests an image classification model. This can be used in order to transform any non-labeled data into labeled data due to its high accuracy, thus converting the task from semi-supervised learning into supervised learning.

The second part of the project is divided into three subcategories of recommendation engine -

1) The first recommendation engine takes in images from the user and recommends products solely based on image similarity,

2) The second recommendation engines takes the product's title and recommends products that has a high similarity score based on the title

3) The final recommendation engine takes into account the title and brand information and recommends products that have a high similarity score based on the title and brand name.

As image-based recommendation is extremely time-consuming for real-time data, and as such, not feasible for deployment, we have restricted it to a static scenario. The images in the static scenario represent a fashion dataset (one of the top categories on Amazon) and are based on 5 categories/types of apparel scraped from Amazon. The categories are hats, watches, shirts, shoes and trousers.

2. LITERATURE SURVEY

2.1 Existing problem

- Limited Discoverability: Customers miss out on discovering new products or items that they may be interested in but are not aware of. A recommendation system helps expose customers to a broader range of options and enhances discoverability.
- Inefficient Search Process: Manual searches require customers to input specific keywords or browse through multiple categories, which can be time-consuming and frustrating.
- Decreased Customer Satisfaction: In the absence of personalised recommendations, customers may struggle to find products that truly

meet their needs and preferences. This can result in decreased customer satisfaction and potential loss of business as customers.

- Information Overload: Without a recommendation system, customers are left to navigate vast catalogues of products on their own. This can lead to information overload, making it difficult for customers to find products that align with their preferences and needs.
- Poor Personalization: A lack of a recommendation system means that customers receive generic suggestions or have to manually search for products. This leads to a poor level of personalization, as individual preferences, browsing history, and behaviour are not taken into account.

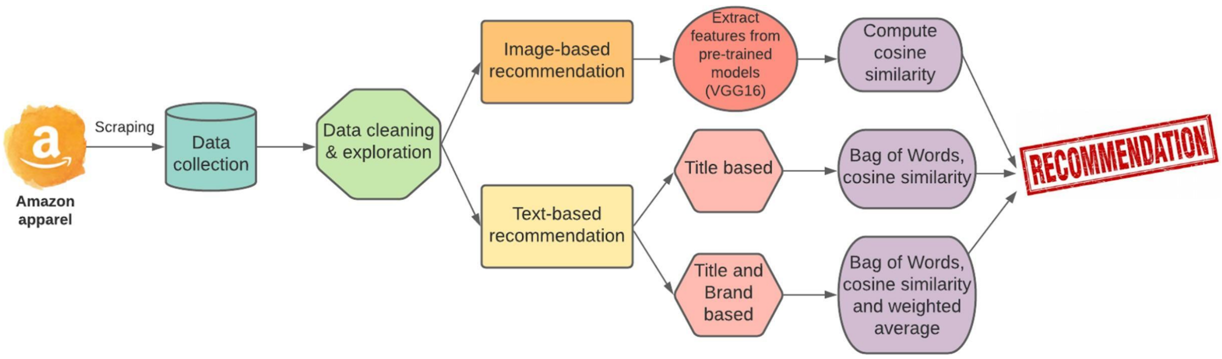
2.2 Proposed solution

We built two recommendation engines in this project. The first type is image based recommendation engine and second type is text based recommendation engine. In order to recommend products based on images, we calculate the cosine similarities between the image features extracted using a pre-trained model. Images were then recommended based on the computed cosine similarities ranked in descending order. The higher the cosine similarity, the more semantically similar the recommended image is to the input image.

In the text based recommendation system, the first method is to recommend products based on the product title and second method is to recommend products based on title and brand of the product.

3. THEORETICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing

- Rapid API
- Python environment
- Pycharm
- Flask
- Input Images

4. EXPERIMENTAL INVESTIGATIONS

Experimental investigations in a product recommendation system project typically involve conducting experiments to evaluate the performance and effectiveness of different recommendation algorithms or approaches. Here are some areas of focus in experimental investigations on PDS:

- Recommendation Algorithm Comparison: Compare the performance of different recommendation algorithms to identify the most effective one for your specific use case. This can involve evaluating algorithms like collaborative filtering, content-based filtering, matrix factorization, or hybrid approaches. Measure metrics such as accuracy, precision, recall, or mean average precision (MAP) to assess the algorithm's performance.

- Evaluation Metrics: Experiment with various evaluation metrics to measure the quality of your recommendations. Common metrics include precision, recall, F1-score, mean average precision (MAP), normalized discounted cumulative gain (NDCG), or area under the receiver operating characteristic curve (AUC-ROC).

- Data Preprocessing Techniques: Investigate the impact of different data preprocessing techniques on recommendation performance. For example, try different methods for handling missing data, data normalization, feature scaling, or text preprocessing (e.g., tokenization, stop-word removal, stemming) if dealing with textual data.

- Hyperparameter Tuning: Experiment with hyperparameter tuning techniques to optimize the performance of your recommendation models. This can involve techniques such as grid search, random search, or Bayesian optimization to find the best combination of hyperparameters for your models.

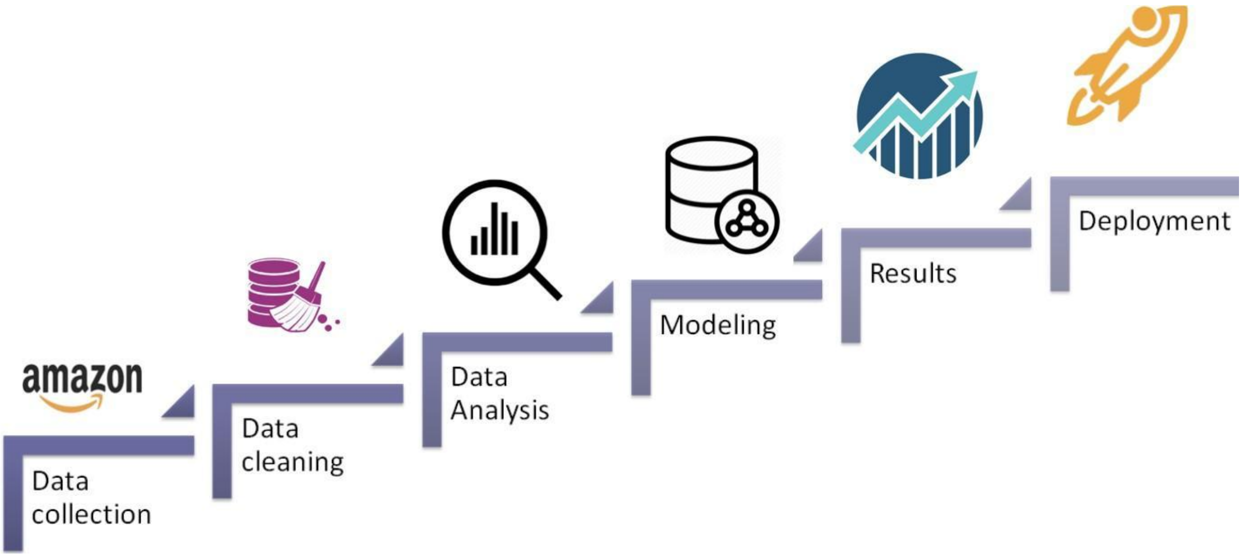
- Cold-Start Problem: Investigate strategies to address the cold-start problem, which occurs when new products or users have limited data available for recommendations. Experiment with techniques like content-based filtering, knowledge-based recommendations, or hybrid methods to mitigate the cold-start problem.

- A/B Testing: Conduct A/B testing to compare the performance of different recommendation strategies in a real-world setting. Randomly assign users to different recommendation groups and measure the impact on metrics like conversion rate, click-through rate, or user engagement.

- Scalability and Performance: Evaluate the scalability and performance of your recommendation system as the dataset size or user base grows. Measure factors like response time, throughput, or resource utilization to ensure your system can handle increasing loads effectively.

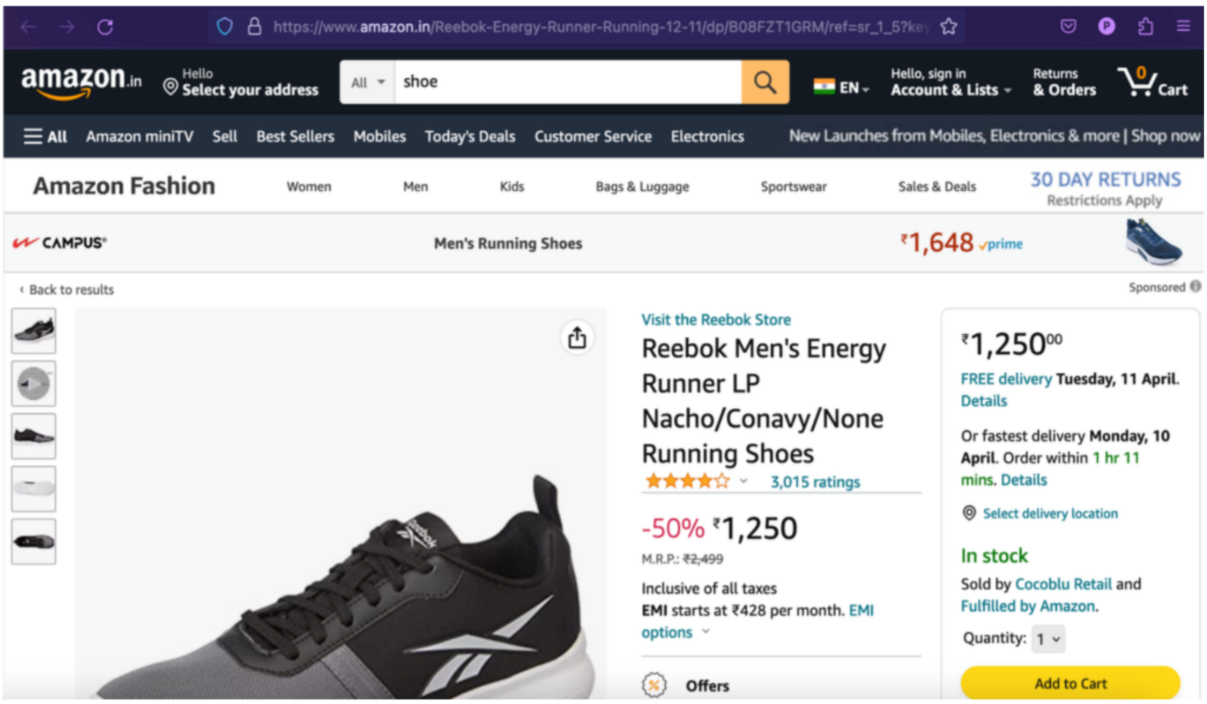
- User Satisfaction and Feedback: Collect user feedback and ratings to measure user satisfaction with the recommendations. Conduct surveys, user interviews, or sentiment analysis to gather qualitative insights and validate the usefulness and relevance of the recommendations.

5. FLOWCHART



6. RESULT

The prototype uses a very simple text-based UI using a single line of text and a single directive to run the python script:
For example:



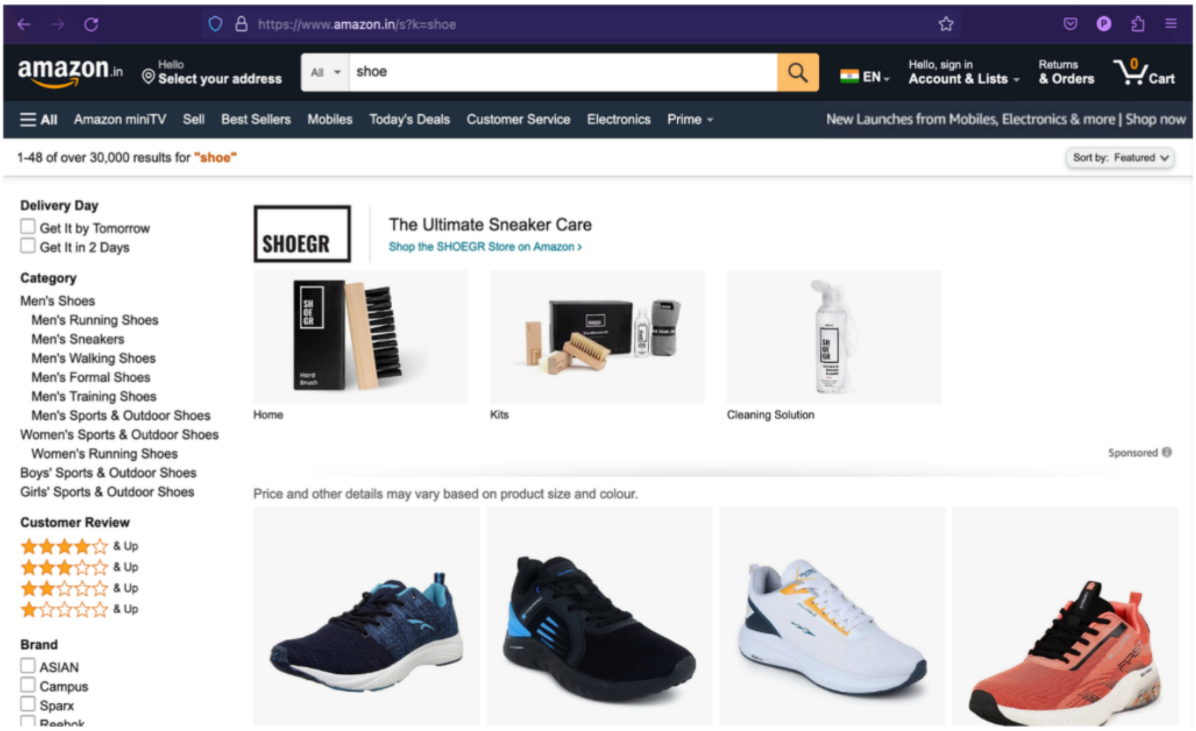
Product of user's choosing on amazon.

```
product_urls.txt
1    https://www.amazon.in/Reebok-Energy-Runner-Running-12-11/dp/B08FZT48FM/ref=sr_1_7?keywords=shoe&qid=1680773744&
```

The user pastes the complete product url into product_urls.txt, a text file within the program files.

```
prahlad@Giltwing amazon-scraper-v3 % python3 product.py --recommend 1 --pages 10
Downloading https://www.amazon.in/s?k=shoe&page=1
Saving Product: Men's Running Shoe
Saving Product: Men's Trend Sports Running Shoe
Saving Product: Shift PRO Sports Shoes - Running, Walking, Gym, Lightweight, Comfort Grip - for Men's & Boy
Saving Product: Men's First Running Shoes
Saving Product: Men's Energy Runner LP Nacho/Conavy/None Running Shoes
Saving Product: Men's OXYFIT (N) Walking Shoe
Saving Product: Mens Sx0500g Running Shoe
Saving Product: Mens 5g-820 Running Shoe
Saving Product: Men's Loire-z126 Running Shoes
Saving Product: Mens Sm 651 Industrial Shoe
Saving Product: Women's Alexa Running Shoes
Saving Product: Men's Crysta Running Shoes
Saving Product: Men's Maxico Running Shoes
Saving Product: Mens Formal Shoes
Saving Product: Tiger Black Lorex Safety Shoes, 8 Inch
```

The user then runs the program using the python directive given above. The program begins running. The number of pages of similar products is specified by the user.



The code automatically extracts the type of product from the url, and searches the same on Amazon. It then extracts individual product details, and stores them in a csv file. Machine learning on the CSV files as per the method specified by the user then returns the top 5 recommended products.

```
Downloading https://www.amazon.in/Reebok-Energy-Runner-Running-12-11/dp/B08FZT48FM/ref=sr_1_7?keywords=shoe&qid=1680773744&sr=8-7&th=1&psc=1
----------------------------------------------------------------------
Original product:
----------------------------------------------------------------------
Product ID :  358
Title :  Men's Energy Runner LP Nacho/Conavy/None Running Shoes
Brand :  Reebok
Price :  1225.0
Reviews :  3015.0
Rating :  4.0


----------------------------------------------------------------------
Most similar products:
----------------------------------------------------------------------

Product ID :  4
Title :  Men's Energy Runner LP Nacho/Conavy/None Running Shoes
Brand :  Reebok
Price :  1250.0
Reviews :  3015.0
Rating :  4.0
Similarity score :  0.9999999999999999

Product ID :  269
Title :  Men's Running Shoes
Brand :  ASIAN
Price :  nan
Reviews :  6816.0
Rating :  3.8
Similarity score :  0.6123724356957945

Product ID :  167
Title :  Men's Running Shoes
Brand :  Klepe
Price :  599.0
Reviews :  193.0
Rating :  3.8
Similarity score :  0.6123724356957945

Product ID :  24
Title :  Men's Running Shoes
Brand :  Campus
Price :  494.0
Reviews :  3039.0
Rating :  4.1
Similarity score :  0.6123724356957945

Product ID :  53
Title :  Men's Running Shoes
Brand :  Sparx
Price :  934.0
Reviews :  15617.0
Rating :  4.0
Similarity score :  0.6123724356957945
prahlad@Giltwing amazon-scraper-v3 %
```

The top 5 recommended products are displayed. This particular recommendation method (1) uses similarity in product title as a basis to recommend products to the user. If the value of argument 'recommend' is set to 2 by the user while running the program, the brand is also taken into consideration. Products of the same brand are recommended at higher priority.

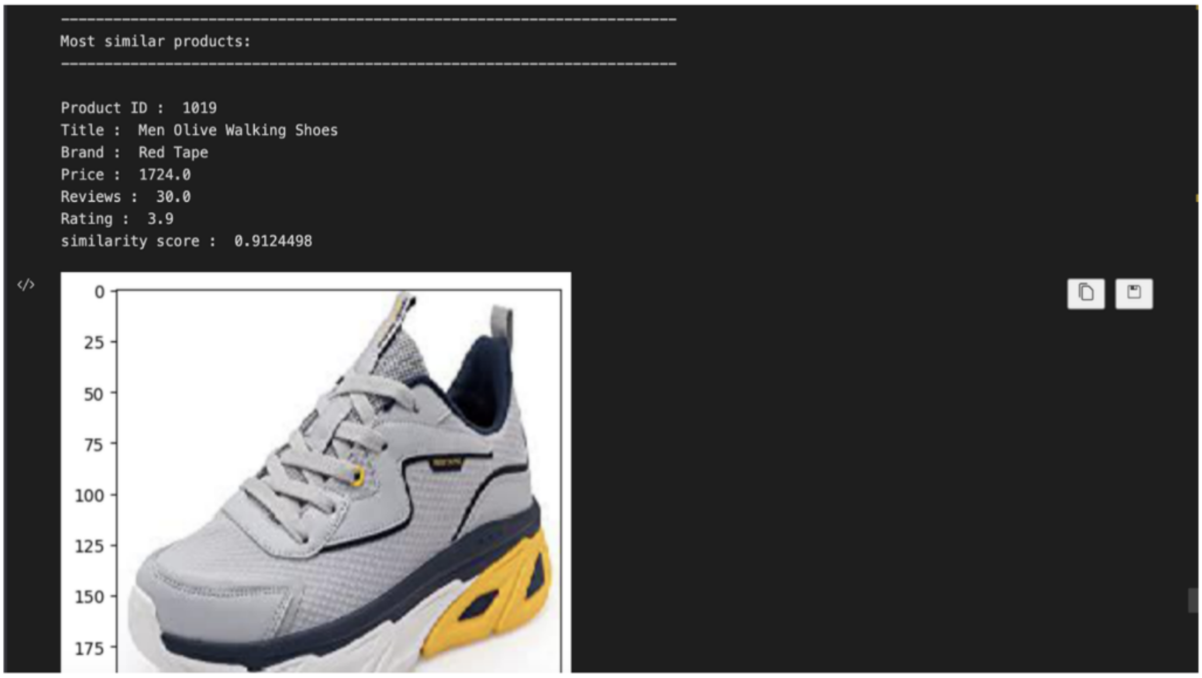
Static scenario:

```python
image_recommend_1(1000,5)
```

Python

```
----------------------------------------------------------------------
Original product:
----------------------------------------------------------------------

Product ID :  1000
Title :  Men Black Walking Shoes
Brand :  Red Tape
Price :  1724.0
Reviews :  2.0
Rating :  4.5
```



An image-based similar product recommendation system was also developed, but runs very slowly due to the large amounts of data and the complexity of the image feature extractor. Thus, it is not deployed.

In conclusion, the product recommendation system helps the user in choosing products suitable to their needs and does so in a unique fashion, providing the user with a much faster method to find similar products using Machine Learning techniques including Natural Language Processing, Deep Learning and Image Processing.

The product recommendation system works as a cohesive package, requiring only minimum details from the user. It acquires data (web scraping), cleans the data (data preprocessing), trains a model (machine learning), and tests the model (product recommendation) automatically and seamlessly, It is thus suitable for deployment.

7. ADVANTAGES & DISADVANTAGES

Advantages:

- Personalised Recommendations
- Increased Sales and Revenue
- Enhanced User Engagement
- Improved Customer Experience
- Discovery of New Products

Disadvantages:

- Accuracy Limitations
- Over Reliance on Popular Items
- Maintenance
- Cost and Infrastructure Requirements
- Limited Understanding of Context

8. APPLICATIONS

- E-commerce Platforms
- Streaming Services
- Social Media Platforms
- Online Travel and Hospitality
- Food Delivery Services
- Online Marketplaces
- Fashion and Apparel
- Online Grocery Stores
- Financial Services
- News and Content Aggregation.
- Gaming Platforms
- Beauty and Cosmetics
- Health and Fitness
- Home Decor and Furniture
- Automotive Industry

9. CONCLUSION

We performed image classification and product recommendation based on images and text in our project. Although the results produced from the image based recommendation system yielded the best results, it is greatly limited by its model complexity which limits its ability to return recommendations quickly to users.

In conclusion, the product recommendation system helps the user in choosing products suitable to their needs and does so in a unique fashion, providing the user with a much faster method to find similar products using Machine Learning techniques including Natural Language Processing, Deep Learning and Image Processing.

The product recommendation system works as a cohesive package, requiring only minimum details from the user. It acquires data (web scraping), cleans the data (data preprocessing), trains a model (machine learning), and tests the model (product recommendation) automatically and seamlessly, It is thus suitable for deployment.

10. FUTURE SCOPE

Work we would like to explore in the future: A recommendation engine that streamlines a fast image-based recommendation, and deployment of the same on the web using Flask (or similar service) for testing and debugging purposes.