WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

"Smart Suggestions, everytime you purchase powered by KNN"

Prof. Sneha Patange[1], Deeksha Shetty[2], Shriyash Kumbhar[3], Smita Bhoine[4], Tanuj Rane[5]

Computer Department Vidyalankar Polytechnic, Mumbai, India.

ABSTRACT : -

This research focuses on enhancing product recommendations for an ecommerce platform dedicated to agricultural products using KNN algorithm. The platform is designed to streamline the purchasing process for both farmers and sellers. While, the website facilitates features such as an integrated chatbot where user can ask any doubts related to agricultural practices powered by 'gemini-pro'. The core of this study lies in optimizing product recommendations based on user's purchases. This approach not only enhances user engagement but also contributes to business growth.

Keyword – KNN, Chatbot

INTRODUCTION :

The primary aim is to create simple, efficient access for the farming population to basic commodities by offering a simple interface of online shopping even for those who have limited digital awareness. This e-commerce platform aims to provide an all-inclusive platform for selling agricultural products. One of the most notable features of the platform is its recommendation system, which uses the K-Nearest Neighbors (KNN) algorithm to analyze user behavior and provide relevant product recommendations. Among various recommendation techniques, the KNN algorithm stands out due to its simplicity, and effectiveness. This algorithm trains itself everytime the endpoint gets hit by the user. The dataset of model gets updated everytime a user places an order. So,it gets trained everytime and gives suggestion. It is a real-time training for dynamic suggestions. This implementation is lightweight yet powerful solution. It's ability to work with small datasets while maintaining high recommendation accuracy makes it an ideal choice for our e-commerce platform.

This personalized approach will help farmers discover products that best suit their needs, improving their overall shopping experience and efficiency.

LITERATURE REVIEW :

This study explores the development of an e-commerce platform that digitalizes agricultural product retail, integrating machine learning techniques to enhance user experience. A significant gap identified in this domain is the absence of a dedicated digital platform that enables farmers to efficiently buy and sell goods while leveraging advanced technologies such as AI-powered recommendation systems.

Existing research highlights key factors that contribute to the success of e-commerce platforms, including website usability, user trust, customer satisfaction, and personalized recommendation systems. However, current solutions often lack specialized recommendation mechanisms designed for agricultural product purchases. To address this, our project implements a K-Nearest Neighbors (KNN)-based recommendation system, which analyzes user purchasing behavior to suggest the most relevant products. By employing KNN, the platform ensures that farmers receive appropriate recommendations based on their needs, thereby improving decision-making and enhancing their shopping experience [1]. Beyond facilitating transactions, the platform integrates an AI-powered chatbot to assist users with queries related to agricultural practices, making knowledge sharing more accessible. By adapting proven e-commerce models and incorporating machine learning-driven recommendations, this project aims to create a user-centric platform.

Our approach is to reduce the technological developmental gap between customers and sellers from rural areas. We want to introduce this AI based recommendation system in local platform which will make people from rural areas familiar with this technology. Our approach serves as an initial step toward AI driven recommendations for smaller businesses, making intelligent e-commerce solutions widely available.

SYSTEM ARCHITECTURE :

The platform follows a muti-tier architecture that integrated a frontend UI, backend processing layer,database for data storage.

- **Frontend (User interface)**

Developed using HTML,CSS and Javascript.

Communicates to backend via jinja templates

- **Backend**

Built using Django for data processing

- **Database**

Stores user profiles,cart details,order details and blogs.
Uses sqlite for structured data

- **Authentication**

Supports CSRF token for authentication for secure user sessions.

Data Flow and communication :

- **User authentication :**

The user is registered and logged in through (CSRF token) django's authentication module.

- **User request handling :**

The frontend send an api request for respected functionality and the backend fetched data from the database and the response is sent back to frontend via jinja templates integrated in HTML page.

- **The recommendation system workflow :**

User activity (purchased products) is saved in database, then system applies KNN algorithm and generate suggestions

IMPLEMENTATION :

The recommendation system in this website is implemented using KNN algorithm . The main aim is to provide personalized and accurate recommendation to users.

Dataset preparation :

A schema is created consisting of user and product. Where user will be foreignkey, and product will be also a foreign key to the respective user model and product model.

Userproduct schema :

```
class UserPurchase(models.Model):
   user = models.ForeignKey(User,on_delete=models.CASCADE)
   product = models.ForeignKey('store.Product',on_delete=models.CASCADE)
```

This schema gets updated everytime when user places an order as you can see below :

This is a view:

```
def billing_process(request):

   if request.POST:
    userid = request.user.id
    cart = Cart(request)
    cart_products = cart.get_prods()
    cart_qty = cart.get_qty()
    amount = cart.get_amount()
    my_shipping = request.POST
    request.session['my_shipping'] = my_shipping
    print(my_shipping)
    for l in cart_products:
      product_id = l.id
      for k,v in cart_qty.items():
        key = int(k)
        if key == l.id:
          pro = Product.objects.get(id=product_id)
          purch = UserPurchase(user=request.user,product=pro)
          purch.save()
          print("purchase saved")
```

Technical details:

Implementation of knn

- Then a user-product matrix is created using which will have user id and product id .
- Then the duplicates (where a user must have purchased same item twice) such records are removed.
- Then a column is added to this matrix (as interaction=1) to every row.
- **Now, a final matrix will be created consisting user,product and interaction. (interaction will be 0 where user have not purchased the product) as you can see below :**

product_id	1	2	3	4	5	6	7	8	18
user_id									
2	1	1	0	1	1	1	0	0	0
7	0	1	0	1	0	0	0	0	0
8	1	1	1	1	0	1	0	0	0
9	0	0	0	0	0	1	1	0	1
10	0	0	1	1	0	0	0	0	0
11	1	0	0	0	0	0	0	0	0
12	1	1	1	1	0	1	0	0	1
13	1	0	1	1	1	0	0	1	0
14	0	0	0	1	0	0	0	0	0
16	1	1	1	1	0	1	0	0	0
17	1	1	0	0	0	0	0	0	0
19	0	1	0	1	0	0	0	0	0

- **Then this matrix is converted to array using numpy as the algorithm is imported from sci-kit learn which required input as an array, as you can see following coding snippet:**

```
user_product_matrix = user_product_matrix.to_numpy()
```

- **Then n_neighbors is passed as 3 and metric cosine is passed as parameters to algorithm as you can see in code snippet :**

```
knn = NearestNeighbors(n_neighbors=3,metric='cosine')
knn.fit(user_product_matrix)
```

- **Now, this function returns knn model,user product matrix and dataframe,as you can see in following code snippet :**

```
return user_product_matrix,knn,df
```

- **Another function is define to return the recommended products, which takes user id as parameter**

```
def get_recommendation(user_id):
```

- **now,knn computes the distance between passed user id and all the other users.**

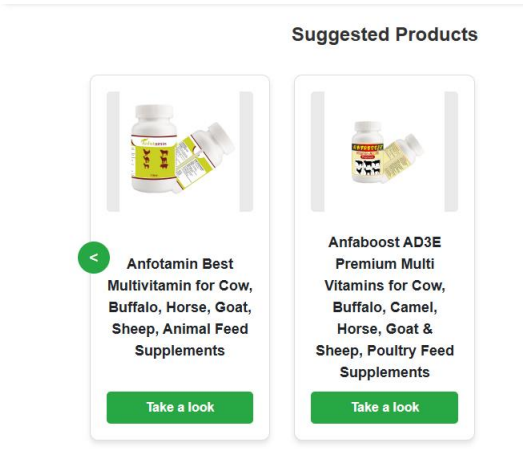
```
distances,indices = knn.kneighbors(user_product_matrix[matrix_index].reshape(1,-1))
```

- **Now, it will suggest the products bought by 3 closest neighbors based on the calculated distance as you can see below :**

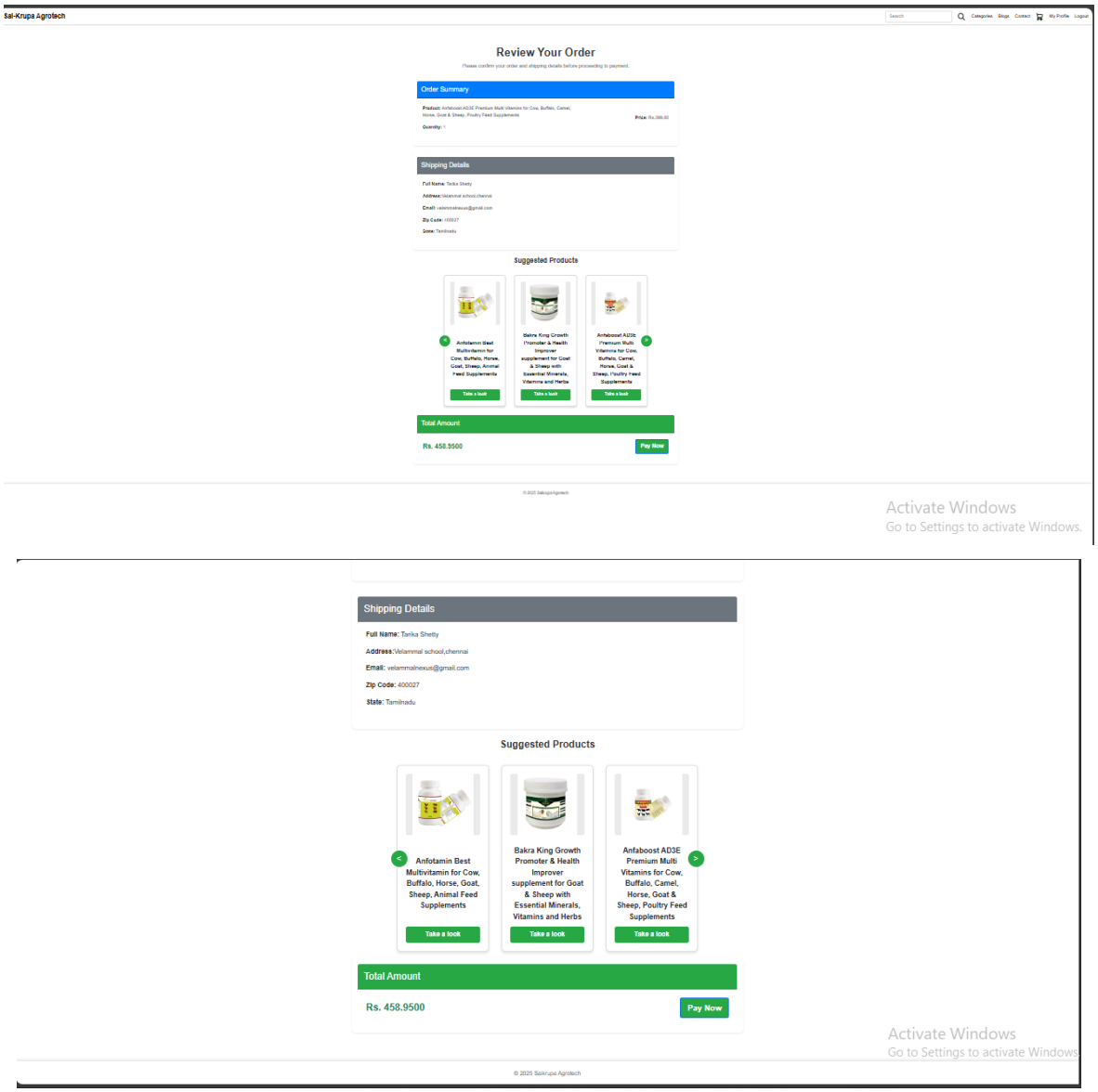
```
[[2.22044605e-16 2.22044605e-16 2.92893219e-01]]
```

- **Hence,the products purchased by them are recommended to the current user as it also returns the indices of those specific users and the products purchased by them can be tracked by their IDs as you can see below :**

```
[[ 1 11  8]]
```



Suggested products page :



CONCLUSION :

The approach of integrating a KNN-based recommendation system, will help users (e.g.farmers) choose the right products as per their needs. Personalized recommendations with higher accuracy are more effective than those generated through traditional filtering methods. As a result, users receive better suggestions, improving user engagement, which will ultimately benefit the stakeholder.

REFERENCES :

1. https://pmc.ncbi.nlm.nih.gov/articles/PMC9388237/