**CHAPTER 1**

**INTRODUCTION**

**1.1General**

Recommendation System is a subclass of information filtering system. They are software agents that elicit the interests and preferences of individual consumers and make recommendations accordingly. They reduce information overload by estimating relevance. They have the potential to improve and support the quality of the decisions consumers make, while searching for products online. It is used to provide suggestions for items to be of use to a user. Instead of a static experience in which user’s search for and buy products, the RS increases the interaction to provide a richer experience. It is directed for individuals who lack sufficient personal experience to evaluate the overwhelming number of alternative items, that a website can offer. It supports users’ in their decision making. From the customer perspective, the RS finds things that are interesting, narrows down the set of choices, helps explore the space of options and helps in discovering new things. On the other hand, from the seller perspective RS helps in increasing customer trust and loyalty, increases opportunities for promotion and persuasion along with obtaining more knowledge about the customers. According to a mathematical approach RS can be seen as a function, given, the user model ( ratings ,preferences, demographics, situational context) and items(with or without description of item characteristics), it will find the relevance score which is used for ranking and finally recommend items that are assumed to be relevant. The Recommendation System appears on most of the e-commerce sites because it provides better conversion rates. There are mainly two types of Recommendation Systems found in e-commerce sites namely Personalized RS (for registered customers) and Non-Personalized RS(for new customers).In RS utility is typically represented by ratings and is intitially defined only on the items previously rated by the users. Therefore the recommendation engine should be able to estimate the ratings of the non-rated items and issue proper recommendations based on these predictions.A typical RS takes either of the two approaches i.e.., Collaborative filtering and content-based filtering. The main goal of a RS is to increase the number of items sold. This goal is achieved primarily because the recommended items are likely to suit the user’s needs and wants. It helps to sell more diverse items and to increase the user satisfaction. It increases the user fidelity and helps to clearly understand what a user wants. In our project we build a personalized Recommendation System ,which collects from the user’s their preferences, which are either explicitly expressed eg: as rating or score or are inferred by interpreting user actions. In order to personalize the recommendation , RS’s exploit a range of information about the user’s. This information can be structured in various ways and the selection of what information to model depends on the recommendation technique. We use three techniques namely collaborative- filtering, content based-filtering and hybrid filtering, which is a mix of both content and collaborative filtering to produce the recommendations.

**1.2 Motivation**

As e-commerce began to develop an urgent need emerged for providing recommendations, derived from filtering the whole range of available alternatives. Users’ were in a dilemma as to what to choose from the immense variety of items that were being offered on these websites. The rapid introduction of e-commerce websites frequently overwhelmed users leading to poor decision making. The availability of choices instead of producing a benefit started to decrease the users’ well-being .It is an understood fact that while choice is good, more choice is not always better. Indeed, choice with itz freedom,autonomy and self-determination can become excessive ,creating a sense that freedom may come to be regarded as a kind of misery inducing tyranny.To resolve this problem of information overload RS was introduced.RS aggregates the online behavior of many people to find trends and make recommendations based on them. This is done by mathematically calculating how one person’s actions relate to another. The success of items that recommend is now growing. RS is no longer a novelty. They are being built for almost every domain where we can give recommendations. Although many different approach to RS’s have been built in the last few years, the interest in this area still remains very high.This is due to growing demand on practical applications ,which are able to provide personalized recommendations and deal with the issue of information overload.RS’s are usually classified according to their approach to rating estimation.Almost all the e-commerce sites have implemented this and that has served as our motivation to develop this project. Apart from the traditional collaborative and content based filtering, in our project we have developed a new hybrid algorithm for giving recommendations. This is aimed to increase the accuracy of our predictions. Depending on the domain and data characteristics we have chosen the mixed hybridization technique for our project.

**1.3 Problem description**

Recommendation systems are currently being used in most of the e-commerce sites. But, in our project we have focused on the processed food items. We are using a hybrid approach, which is a combination of content based and collaborative filtering for implementing the recommendation system. They can be implemented by having each method , make the prediction on its own and then the results are merged. This helps increase the accuracy and diversity of predictions. Implementing a hybrid algorithm helps very clearly to achieve the fundamental goal of RS i.e,.. to expose user’s to novel content that they would not have seen otherwise.

**1.4 Related Work**

Currently Recommendation systems play an important role in many of the applications in WWW. Most of the e-commerce websites have implemented this and have been hugely successful. One of the most popular recommender systems is that of Amazon’s.They make heavy use of item-item collaborative filtering approach.For each item X, amazon builds a neighborhood of related items Y.It is simpler to implement and is faster to update recommendations.They first associate each item with the list of users’ who have brought/looked at it.The similarity between any two items could then be a normalized measure of the number of users’ they have in common or the cosine distance between two items. Pandora and last.fm are other two sites that have successfully implemented the recommendation system. Pandora’s recommendation systems are based on the inherent qualities of music. If we give Pandora an artist or song, it will find similar music in terms of melody, harmony ,lyrics ,vocal character and so on.Pandora likes to call these musical attributes “genes” and it’s database of songs classified against hundereds of such attributes.Last.fm is a social recommender. It assumes that if a person and a group of other people enjoy many of the same artists, then that person will probably enjoy other artists popular with that group.Last.fm provides users an optional plug-in that automatically monitors their media player software so, whatever they listen to, whether it came from last.fm or not-can be added into their last.fm profile and can be used as a basis for recommendation. The itunes music store from apple has a “just for you” feature that recommends songs and albums.Netflix.com recommends movies and tv shows to watch. Applications such as digg.com also draw on the same idea of aggregating the actions of many people to the present content. Reddit harness users actions to recommend top news stories. The new york times has a “most popular” section that displays the most emailed and most blogged stories.Youtube is another most famous application that uses recommendation systems. They provide personalized video recommendation to its users.It draws its data from two main sources. The first is content data like the meta data like titles and descriptions. The second is user activity data ,which is categorized as explicit attributes such as ratings and favorites and implicit attribute such as view time.The most popular social networking site, Facebook also uses recommendation techniques to suggest friends. They use multiple methods to suggest friends like people who are in same groups, people who have liked similar pages, mutual friends of a users’ friends. Etc.. Of late, Facebook has started recommending people who check in at the same locations. By the advent of these recommendation systems, it has become much easier for the users’ to take decisions in the virtual world.

**1.4.1 System Requirements**

Operating system : Windows

Memory :256 MB Ram

Hard disk capacity: 40 Gb

CPU : Pentium IV 2.66 GHZ

**CHAPTER 2**

**OVERVIEW OF THE PROPOSED WORK**

Proposed work

2.1 Introduction of problem and its related concepts

We have developed a RS for processed food review using a hybrid approach which is a combination of both content based and collaborative filtering. The content based and the collaborative based filtering is first implemented and then the results are merged. In content based filtering , the items are recommended to a user based on a comparison between the content of the item and the user profile.The user profile is built based on the features of the items previously rated by that user.The recommendation process basically consists in matching up the attributes of the user profile against the attributes of the content object. The result is a relevance judgment that represents the user’s level of interest in that object.In content based systems, the recommendation process is performed in three steps each of which is handled by a separate component.

* Content analyzer

The main responsibility of this component is to represent the content of items (e.g., documents, web pages , product descriptions etc..) coming from information sources in a form suitable for the next processing steps.

* Profile learner

This module collects data representative of the user preferences and tries to generalize this data ,in order to construct the user profile. The generalization strategy is realized through machine learning techniques , which are able to infer a model of user interests starting from items liked or disliked in the past.

* Filtering component

This module exploits the user profile to suggest relevant items by matching the profile representation against that of the items to be recommended. The result is a binary or continuous relevance judgment.

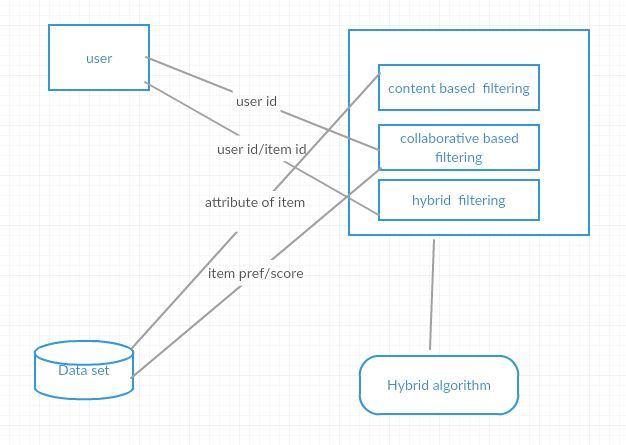
Collaborative filtering is a method of making automatic predictions about the interests of a user by collecting preferences or information from many users. The motivation for collaborative filtering stems from the idea that people often get the best recommendation or suggestion from someone with similar taste to themselves.CF explores techniques for matching people with similar interest and making recommendations on this basis. CF requires user’s active participation, an easy way to represent users’ interests into the system and algorithms that are able to match people with similar interests. Typically, the workflow of a collaborative filtering system is :

1. A user expresses his or her preference by rating items of the system. These ratings can be viewed as an approximate representation of the users’ interest in the corresponding domain.
2. The system matches this users’ ratings against other users and finds the people with most similar taste.
3. The system recommends items that the similar users’ have rated highly but not yet being rated by this user.

There are mainly two types of collaborative filtering namely, user based and item based. In user based CF the system looks for users’ who share the same rating pattern with the active user and use the ratings from those like-minded users’ to calculate the prediction for the active user. The item based CF proceeds in an item-centric manner. The system builds an item –item matrix determining relationships between pairs of items, and it infers the taste of the active user by examining the matrix and matching the users’ data.The hybrid recommendation is a very promising approach, which can provide with a lot of synergies compared to the basic recommendation algorithms.There are mainly four types of recommendation techniques , which are used for constructing hybrids namely collaborative-filtering, content based , demographic and knowledge based. In our project we have used the combination of content based and collaborative along with a popularity based approach for building the hybrid recommendation system.

2.2 Overview of the proposed system

Through our project we are trying to develop a new personalized hybrid recommendation system to display the item preference and item ranking by combining the content based and collaborative-based filtering. The hybrid approach combines two types of information while it is also possible to use the recommendations of the filtering techniques independently.The hybrid recommendation systems are generally classified into seven categories namely weighted,switching,mixed,feature combination,feature augmentation,cascade and meta level. We have implemented the mixed hybrid recommender which is based on merging and presentation of multiple ranked lists into one.Each component of this hybrid is able to produce a recommendation list and the core algorithm of mixed hybrid merges them into a single list.So the users who logs into our website will get three separate class of recommendations namely content-based , collaborative and hybrid which is an amalgamation of both. Through this we try to predict the most suitable product or services for a user.



We have three main modules in the system namely content based, collaborative and hybrid. We have the dataset which comprises of all the user data like the product id, user id,profile name,helpfulness,score,time of review,summary and the review text.For content based filtering we consider the attribute of the item i.e.., the score which will be taken from the dataset. In collaborative filtering, we look for users who share the same rating pattern with the active user.Those ratings from the similar users will be used to make predictions for the active user. In hybrid filtering a cosine similarity matrix is used to find the neighboring user along with a popularity based model , content-based and collaborative filtering techniques. The availability of three sets of recommendations makes it very easier for the user to choose the right product or service that would meet his/her needs and requirements.Thus our system explicitly predicts the users’ preferences.

**CHAPTER 3**

**ANALYSIS AND DESIGN**

**3.1 Brief Introduction**

A recommendation system is an intelligent system which compares the existing data of the user from the dataset to predict the possible liking for that user. There are different ways of filtering. They are Collaborate Filtering, Content Filtering and Hybrid Filtering.We have developed a Web Based Recommendation system which validates the given set of processed food data reviews and compare the traditional method of User-User, Item-Item and User-Item filtering with the hybrid algorithm we derive so as to show the item preference and item ranking. Our Project is working on an efficient way to form a recommendation system for processed food by implementing a new hybrid algorithm.

**3.2 Requirement Analysis**

**3.2.1 Functional Requirements**

* Predefined score is mandatory for content based filtering.
* SQL injection shouldn’t be there.
* In the sign up form it is mandatory to enter the score.

**Domain Requirements**

* The programming language used should be python.
* Boot strap must be used for the GUI.
* The database used is postgre sql.
* The cloud platform to be implemented must be IBM blue mix

**User requirements**

* Users are not supposed to enter null values.

**3.2.2 Non Functional Requirements**

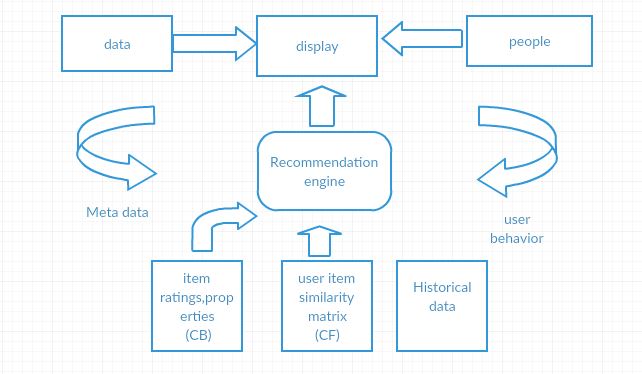
Coverage- The recommendation system that we have developed provides excellent coverage due to the efficiency of algorithms. The coverage includes several distinct properties of the system such as item space coverage and user space coverage. Coverage can be referred as the proportion of items that the recommendation system can recommend (CB) or it can be the proportion of user or user interactions for which the user can recommend items (CF).

Confidence- The confidence in the recommendation system can be defined as the users’ trust in the system’s recommendations or predictions. It is beneficial for the recommendation system to recommend few items that the user already knows and likes. By this way the user observes that the system provides reasonable recommendations , and increases his/ her trust in the system recommendations for unknown items.

Novelty- Novel recommendations are recommendations for items that the user did not know about. An obvious and easy approach is to filter out the items that the user has already rated or used, and recommend other items which match with the user profile i.e, the novelty should be considered only for those items which are relevant to the user. Accuracy is important as irrelevant recommendations may be new to the user but still worthless.

Utility- Most of the e-commerce websites use recommendation system to increase their revenues or utility. In such recommenders , measuring the utility is more important than measuring the accuracy of predictions. User ratings can be used to calculate the utility. In short, the recommendation system should serve it’s primary purpose of increasing the revenue.

**3.3 Detailed design**



**3.4 Module Description**

CONTENT BASED FILTERING

The content based recommendation system has its roots in information retrieval and information filtering research. The main task of content based recommendation is to learn user preferences and to locate/recommend items that are similar to the user preferences. The content based systems are designed mostly to recommend text-based items, the content in these systems is usually described with keywords. For example, a content based component of the fab system , which recommends web pages to users’ represent the web page content with the 100 most important words. In content based recommendation methods, the utility u(c, s) of item s for user c is estimated based on the utilities u(c, si) assigned by user c to items si belongs to S that are similar to item ‘s’. Content based recommendation system recommend an item to a user based upon a profile of the users’ interest and description of the item. All content based recommender systems have few things in common. They are means for description of items, user profiles and techniques to compare profile to items to identify what is the most suitable recommendation for a particular user. In order to determine what items to recommend , collected user information is compared against content features of the items to examine. The system has a huge database consisting of the items to be recommended and the features of these items and it is termed as item profile. The users will provide some sort of information about their preferences to the system. Combining the item information with the user preferences , the system builds a profile of the users. According to the information existing in a target users profile, the system recommends suitable items to the user.To make a better personalized recommendation, we can utilize the features of items and user profiles. An item profile is defined by its important features. For example a book can be described by its title, author, isbn number etc,.. To calculate the similarity between two items, a weighing procedure is used. Depending on the domain, features can be represented either by Boolean values or by a set of restricted values. For instance, imagine we want to analyze chapters in a book about different kinds of topics. A Boolean value could indicate whether a word is contained in a chapter or not, while the integer value will count the number of times the word is appearing in the book. To build a user profile, information of a user can be used. User information might be provided explicitly by the individual person or gathered implicitly by a software agent. There are three main approaches to give explicit relevance feedback:

* like/dislike- Items are classified as ‘relevant’ and or ‘not relevant’ by adopting a simple binary rating scale.
* Ratings- a discrete numeric scale is usually adopted to judge items. Alternatively, symbolic ratings are mapped to a numeric scale.
* Text/comments- Comments about the items are collected and presented to the user as a means of facilitating their decision making process. Some of the prominent example can be the customer feedback at amazon or ebay. It helps the users in understanding, whether an item has been appreciated by the other users or not. Textual comments can be helpful, but the only disadvantage is that it can overload the active user because he/she must read and interpret each and every line to find out whether it is a positive comment or a negative comment.

In order to build the profile of the active user au ,the training set TRA for the active user must be defined. TRa  is the set of pairs (IK, RK) where RK  is the rating provided by the active user on the item representation IK. From the set of item representation labels and ratings, the profile learner applies supervised learning algorithms to generate a predictive model, the user profile, which is stored in a profile repository for later use by the filtering component. From the item representation the filtering components predicts whether it is likely to be of interest for the active user ,by comparing features in the item representation to those in the representation of user preferences. With respect to the user profile, the filtering component will implement some strategies to rank the potentially interesting items. Top ranked items are included in a list of recommendations La, that is presented to au(active user).  It is possible that the changes and preferences of the user change over time, so in that case up-to-date information must be maintained and provided to profile learner in order to automatically update the user profile. Once the feedback is generated, further feedback is generated by letting the users’ state their satisfactions or dissatisfaction’s with the items. Once the feedback is gathered , the learning process is performed again on the new training set and the resulting profile is adapted to the updated user interests. The dynamic nature of user preferences can be understood by the iteration of feedback learning cycle.

COLLABORATIVE FILTERING

Collaborative filtering methods produce user specific recommendations of items based on patterns of rating. It does not need exogenous information about either items or users. In collaborative filtering, to facilitate the comparison between items and users, there are two primary approaches, i.e, neighborhood based approach and latent factor models. In our project we have used the neighborhood approach. It focus on relationship between items or alternatively, between users. In a typical collaborative filtering scenario, there is a list of P users, {u1, u2, ….., up} and a list of Q items, {i1,i2,., iq}

And each user ui has a list of items Iui which the user has rated , or about which their preferences have been inferred through their behaviors. Collaborative filtering algorithms need to have the ability to deal with highly sparse data, to deal with increasing number of users and items and to make satisfactory recommendations in short period. The neighborhood based CF algorithm uses the entire sample of user-item database to generate a prediction. Every user will be part of a group of people with similar interests. By identifying the neighbors of an active user a prediction of preferences on new items for him or her can be produced. This algorithm uses the following steps. Calculating the similarity or weight Wi,j  which reflects the distance or weight between two users or two items I and j, produce a prediction for the active user by taking the weighted average of all the ratings of a user or item on a certain item or user. To generate a top-N recommendations we find K most similar users or items after computing the similarities , then aggregate the neighbors to get the top-N most frequent items as the recommendation. The most critical step in the CF algorithms is the computation of the similarity. In item based CF , the basic idea of similarity computation between two items I and j is to find out users who have rated both of these items and then apply a similarity computation to determine the similarity between the two co-rated items of the users. For a user based CF algorithm we first calculate the similarity between the user’s u and v, who have both rated the same items.

User based top-N recommendation algorithm.

Firstly the k most similar users (nearest neighbors) to the active user are identified. Next a set of items C, purchased by the group together along with their frequency are identified. With the set C, the users based CF techniques, then recommend the top-N most frequent items in C, that the user has not purchased.

Item based top-N recommendation algorithm.

These algorithms firstly compute the k most similar items for each items according to the similarities; then identify the set C, as candidates of recommended items by taking the union of the k most similar items and removing each of the items in the set U, that the user has already purchased; and then calculate the similarities between each item of the set C and set U. The resulting set of items in C ,sorted in decreasing order of similarity will be recommended on the top-N list.

HYBRID FILTERING

The hybrid recommendation systems are a mix of single recommendation systems as sub-components. They were introduced mainly to cope with the problem of conventional recommendation systems such as cold start problem and stability versus plasticity problem. When learning based techniques like collaborative and content based filtering are used, then the cold start problem may occur. The learning stages of these algorithms are based on users’ information. In most cases the user has to input their ratings or preferences manually and in most cases this is very time consuming and is hard to be achieved. Stability or plasticity problem means that sometimes it is hard to change the established users’ profiles, which have been established after a given period of time using the systems. Hybrid recommendation systems can produce outputs, which outperform the single component systems by combining the multiple techniques. The most common hybridization technique is combining or merging different methodologies like mixing CN and CF. Burke classified the hybrid recommendation systems into seven categories namely weighted, switching, mixed, feature combination, feature augmentation, cascade and meta level.

* Weighted- The score of different recommendation components are combined numerically. Therefore, components must be able to produce its recommendation score which are numerically combinable. Also, the components must have consistent relative accuracy across the product space.
* Switching hybrid- The system chooses among the recommendation components and then chooses one. This selection is made according to the situation.
* Mixed hybrid- This is a hybrid which is based on the merging and presentation of multiple ranked lists into one. Each component of this hybrid should be able to produce recommendation lists with ranks and the core algorithm of mixed hybrid merges them into a single ranked list. The issue here is how the new rank scores should be produced.
* Feature combination hybrid- Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
* Feature augmentation hybrid- This is similar to the feature combination hybrids but different in that the contributor generates new features. It is more flexible and adds smaller dimension than feature combination hybrids.
* Cascade hybrid- In this the recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of higher ones.
* Meta level- One recommendation technique is applied and it produces some sort of model, which is then used as an input for the next one.

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 Tools Used**

**General Introduction**

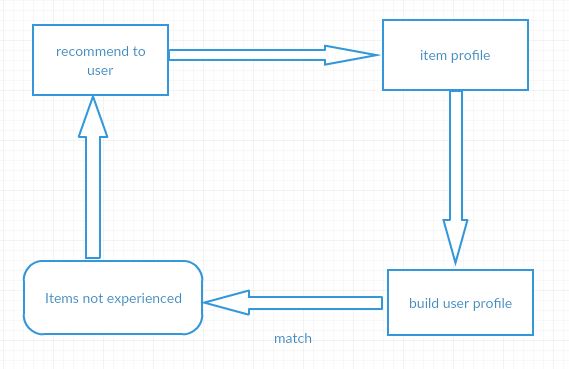
Django is a high level, python web frame work which encourages rapid development and clean pragmatic design. It follows MVC(Model view controller ) architectural pattern. Django’s primary goal is to ease the creation of complex, database-driven websites. It emphasizes reusability and pluggability of components and rapid development. It provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models. It is very fast, reassuringly secure and exceedingly scalable. Django uses a “batteries-included” philosophy. The principle behind batteries included is that the common functionality for building web applications should come with the frame work instead of as separate libraries. For example authentication, url routing, a templating system, an object-relational mapper, and database schema migrations are all included with django framework. Some of the well-known sites that use django are pintrest, instagram, Mozilla. The core django framework consists of an object relational mapper which mediates between data models and a relational database. The other components of the core framework are a light weight and stand-alone web server for development and testing, a form serialization and validation system, a template system that utilizes the concept of inheritance, a caching framework which can use any of the several cache methods, an internal dispatcher system which allows components of an application to communicate events to each other and an interface to python’s built in unit test frame work. The configuration system allows third party code to be plugged into a regular project, provided that it follows the reusable app. The main django distribution also bundles a number of applications in its “contrib” package which includes an extensible authentication system, a dynamic administrative interface, a site frame work that allows one django installation to run multiple websites , each with their own content and built in mitigation for cross site request forgery, cross site scripting, SQL injection, password cracking and other typical web attacks most of them turned on by default. Django can be run in conjunction with Apache, NGINX, using WSGI or Cherokee using flip (a python module). Django officially supports four database back ends namely PostgreSQL, MySQL, SQLlite and oracle.

**Reason for selecting tool**

Security is of paramount importance while building any application and django provides the best features for that. Django protects the users against most of the XSS attacks like cross site scripting. Django has built-in protection against most types of CSRF attacks provided the user has enabled and used it appropriately. CSRF protection works by checking for a nonce in each POST request. This ensures that a malicious user cannot simply “replay” a form POST to the website and have another logged in user unwittingly submit that form. Click jacking is a type of attack where a malicious site wraps another site in a frame. This attack can result in a unsuspecting user being tricked into performing unintended actions on the target site. Django contains click jacking protection in the form of X-frame-options middleware which in a supporting browser can prevent a site from being rendered inside a frame. Django is an open source, low cost, platform agonistic, high performance modern architecture. It is highly modular. It enables easier maintenance and independent development of functional modules with different skill sets. It is being built on standard language and is able to take advantage of libraries developed for other purposes like imaging, graphics, scientific calculations and many more.

**4.2 Methodology**

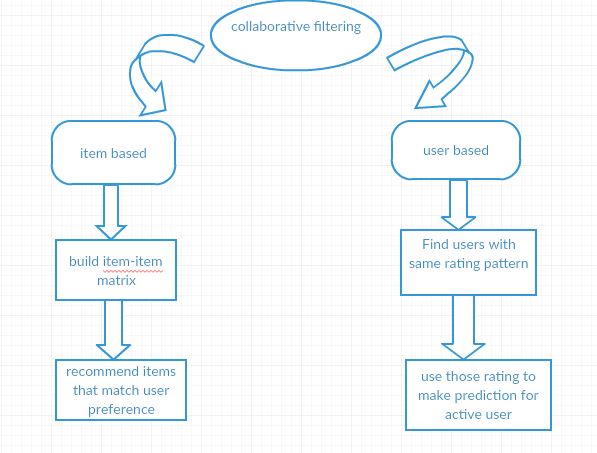
Content based module



Algorithm

* It filters and recommends the item based on the pre-defined attribute.
* In the sign up page users enter their preferred score range for the item.
* From the data those items whose score belongs to the specified range are selected and displayed.

Collaborative filtering



Algorithm

* Items are recommended based on the user preferences.
* When user hits the like button after reading a particular review, then respective item preference and user preference will be stored in the database.
* Those items rated by the preferred user will be recommended to the active user.

Hybrid filtering

Algorithm

* A combination of meta level and mixed hybrid recommendation is used.
* With the help of user id, the item preference and score is taken from the database.
* The scores are added and average is found.
* This score will serve as an input for the content based filtering.
* From the resultant list, the most popular items will be recommended, i.e,, the ones with the highest score

**4.3 Implementation**

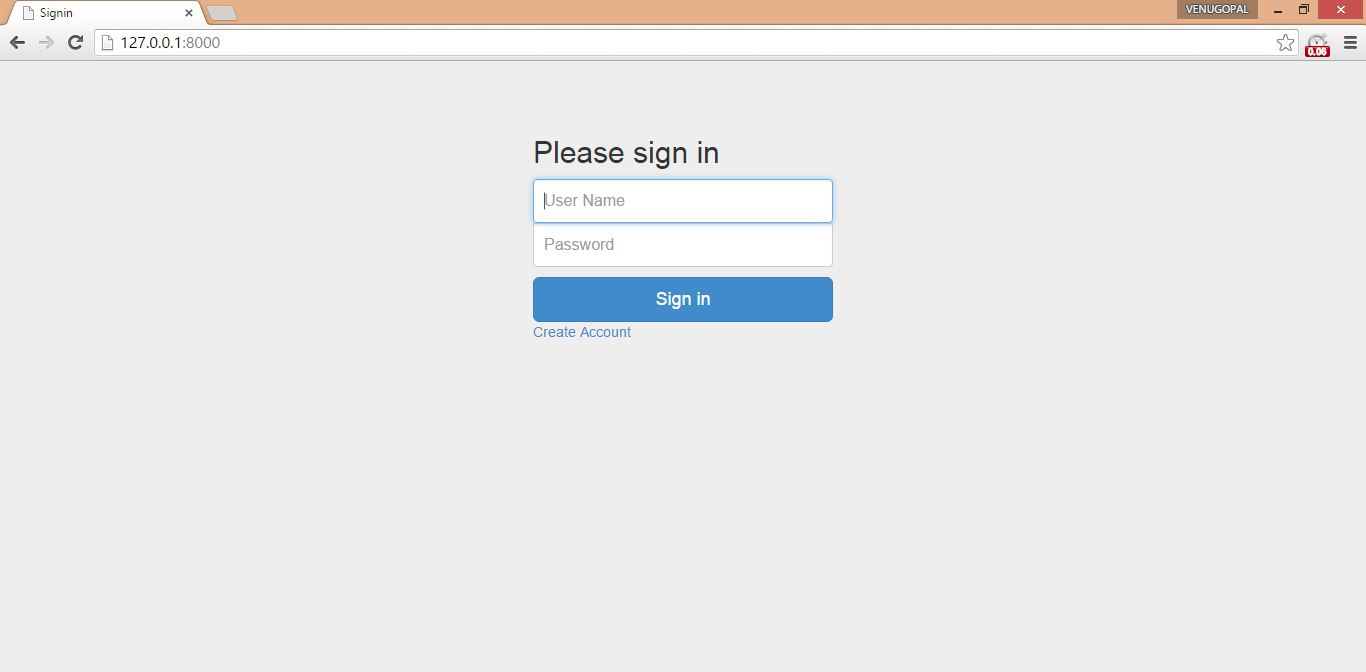
**Graphic User Interface**

The GUI was developed using django framework. There are mainly four different modules in the system namely, sign up module, content based, collaborative and hybrid filtering. The GUI is extremely user friendly with less chaos and ambiguity and the user can easily navigate between the pages. The security protocols in Django adds to the robustness of our application.

**Screen shots and explanation**

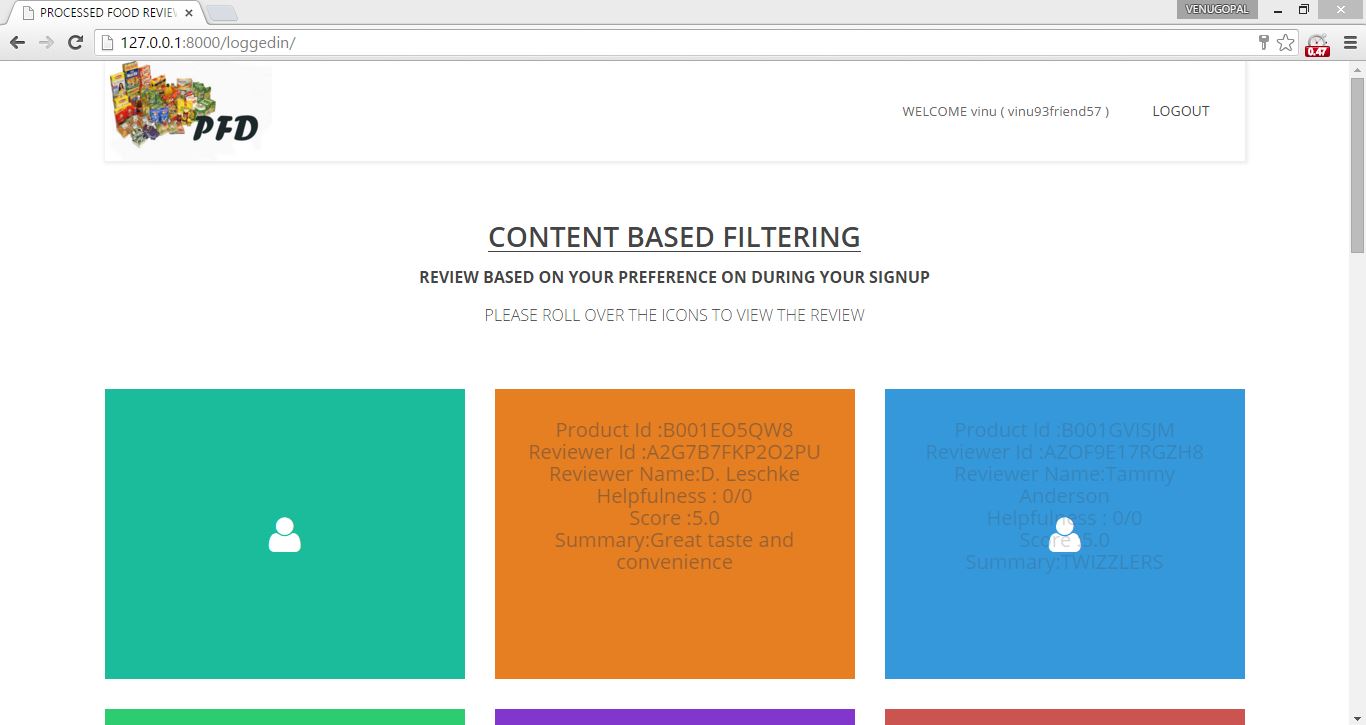
Module 1

1. Sign in module



Explanation

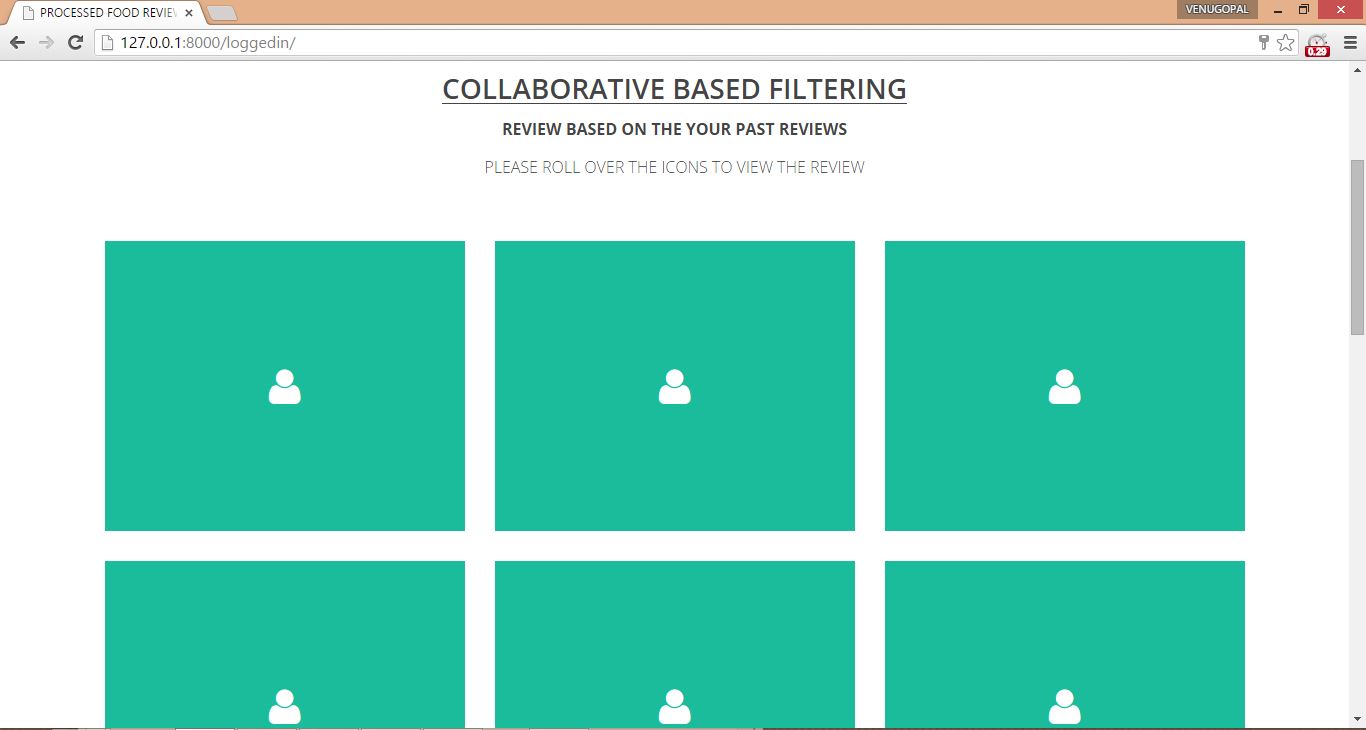
This is the first module in our system. For a user to be able to read the reviews, he/she has to first sign in to the system.For the users who have not yet registered in the system can do so by clicking on the create account option, where the user will be guided as to how to create the account.

2.Content based

Explanation

This is the second module of the system. It is the content based filtering, where the user has to enter a score range of his/her preference during the sign in process itself. Those reviews whose score belongs to that particular range will be displayed. The user can roll over the box to read the review.

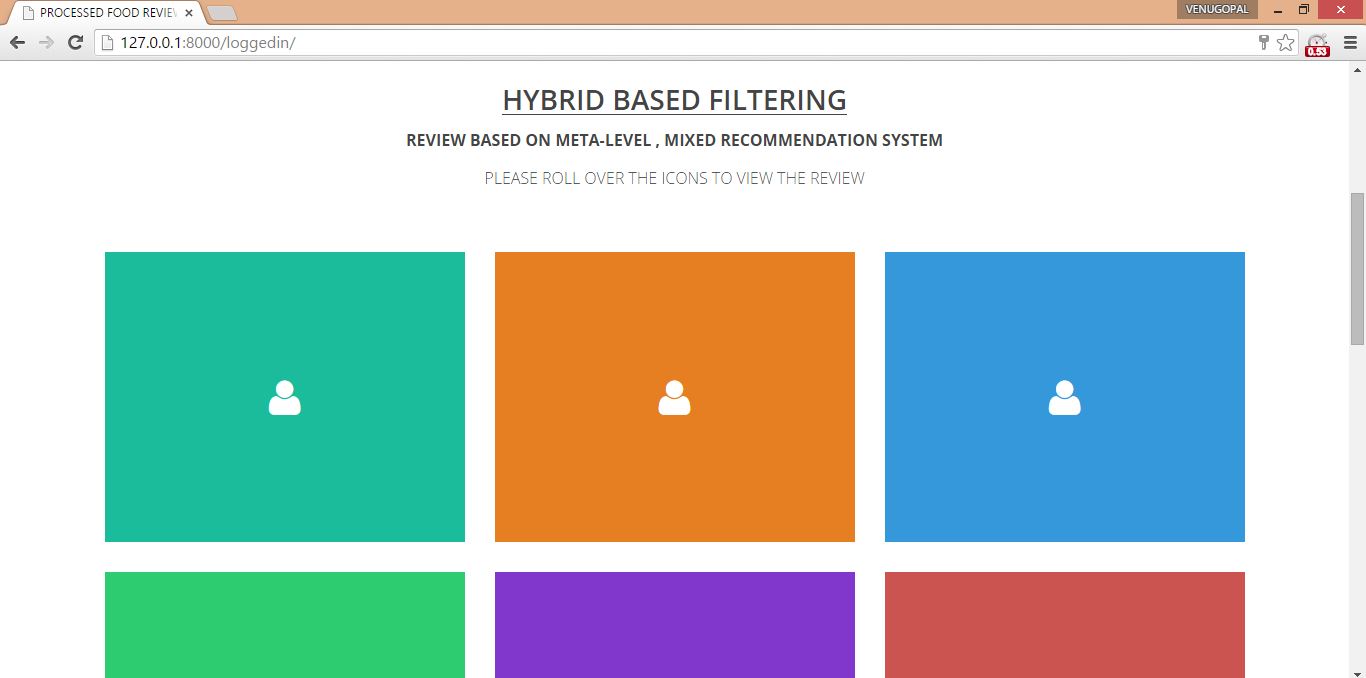
3.Collaborative filtering



Explanation

This is apparently the third module of our system. The collaborative filtering technique works on the principle that , if A likes B and B likes C, then A would also C. Here the recommendation will be done based on the user preferences. When the user presses the ‘like’ button after reading a review, the respective item preference and user preference will be stored in the database. The items liked by the preferred user will be recommended to the active user.

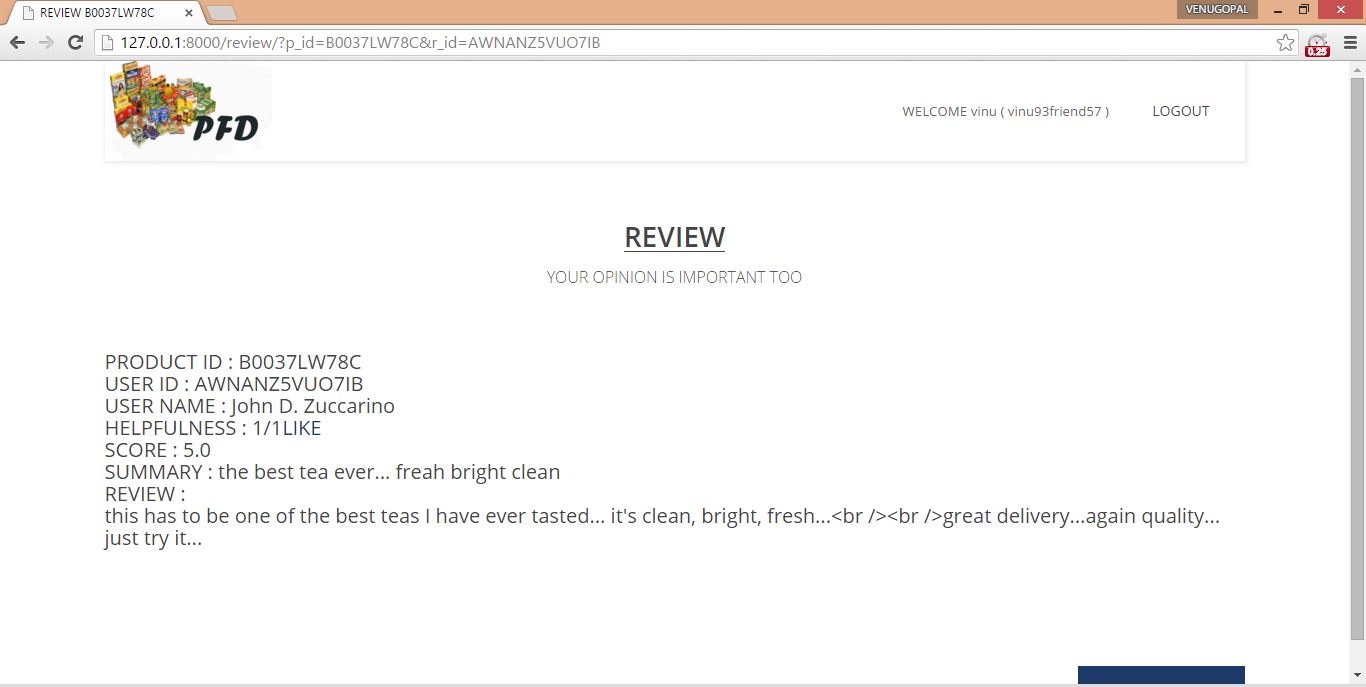
4.Hybrid filtering



Explanation

This is the third module of our recommendation system. In hybrid recommendation, system the review is given based on meta level and mixed recommendation. This is tend to be much more efficient and accurate than the other two recommendation approaches.

Review



Explanation

This is how the review would look like. When the user navigates to the next page,the complete review of the item will be displayed as above.

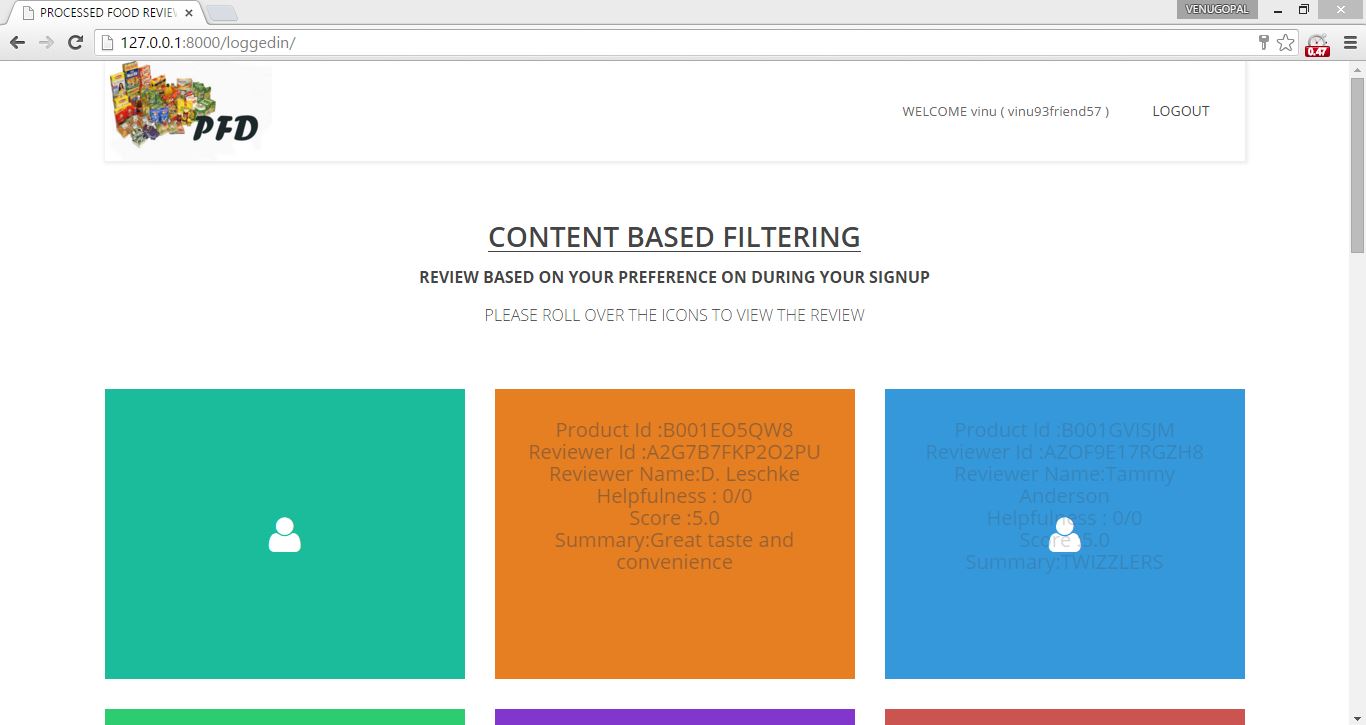
**CHAPTER 5**

**RESULTS AND DISCUSSION**

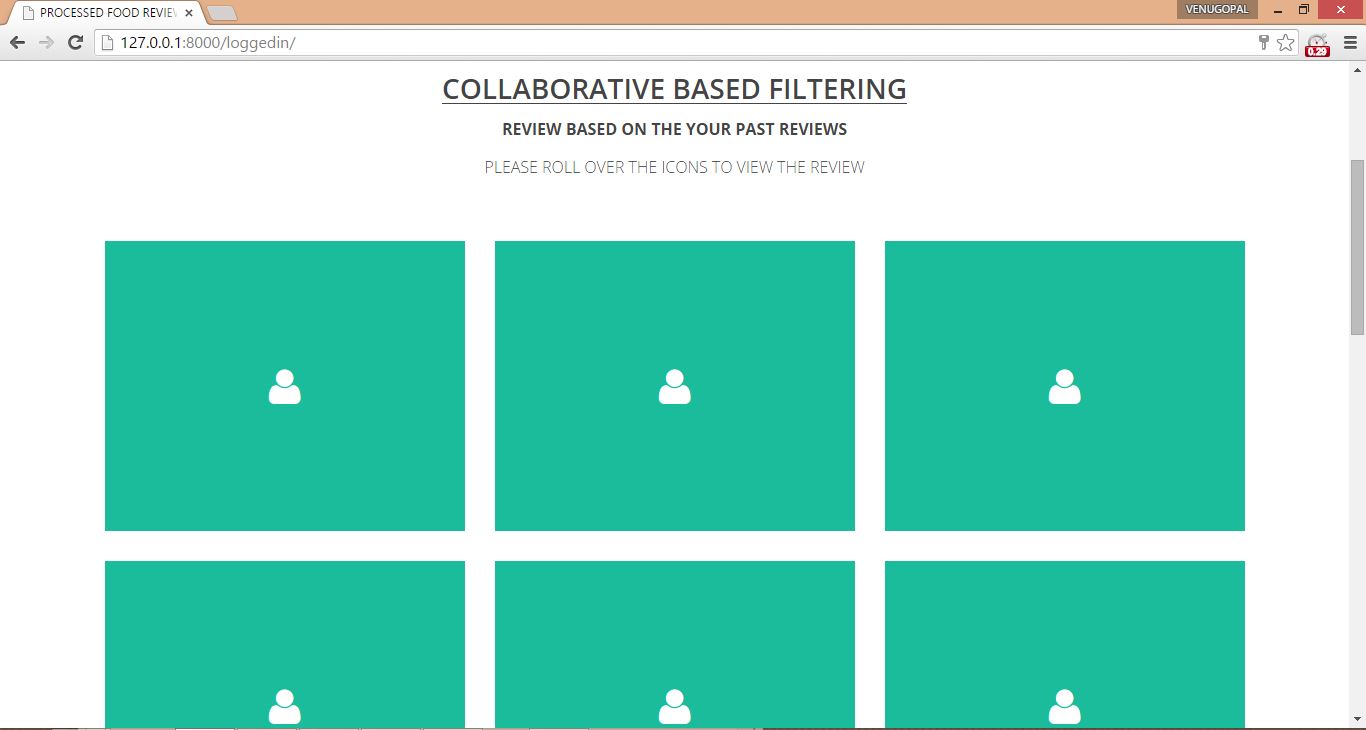
**5.1 Results**

Through our project we have successfully developed a hybrid recommendation system. To an extend we have succeeded in increasing the utility and accuracy of our recommendations with the help of efficient algorithms. We display the recommendations based on all the three filtering techniques ,and it is upto the users’ convenience as to which recommendation to choose. We have succeeded in giving the user diverse options through our implementation. This will help them in making more accurate and efficient choice that will meet their requirements. In an era of tons of fake information in the internet we have tried to increase the robustness of our application by implementing strict security protocols. Our project overcomes some of the possible defects that are found in content and collaborative filtering with the help of our efficient algorithm. In collaborative filtering, item or user cold start is a possible defect as a new user will not have any preference. Similarly in content based we use pre-determined parameter/attribute to get the filtering. In this case, a user whose preference can vary over time may have some issues getting the preferred preference. The algorithm which we designed overcomes this by either displaying content based filtering if collaborative item preference is not given or give a proper preference by assumption by combining the content and collaborative filtering.

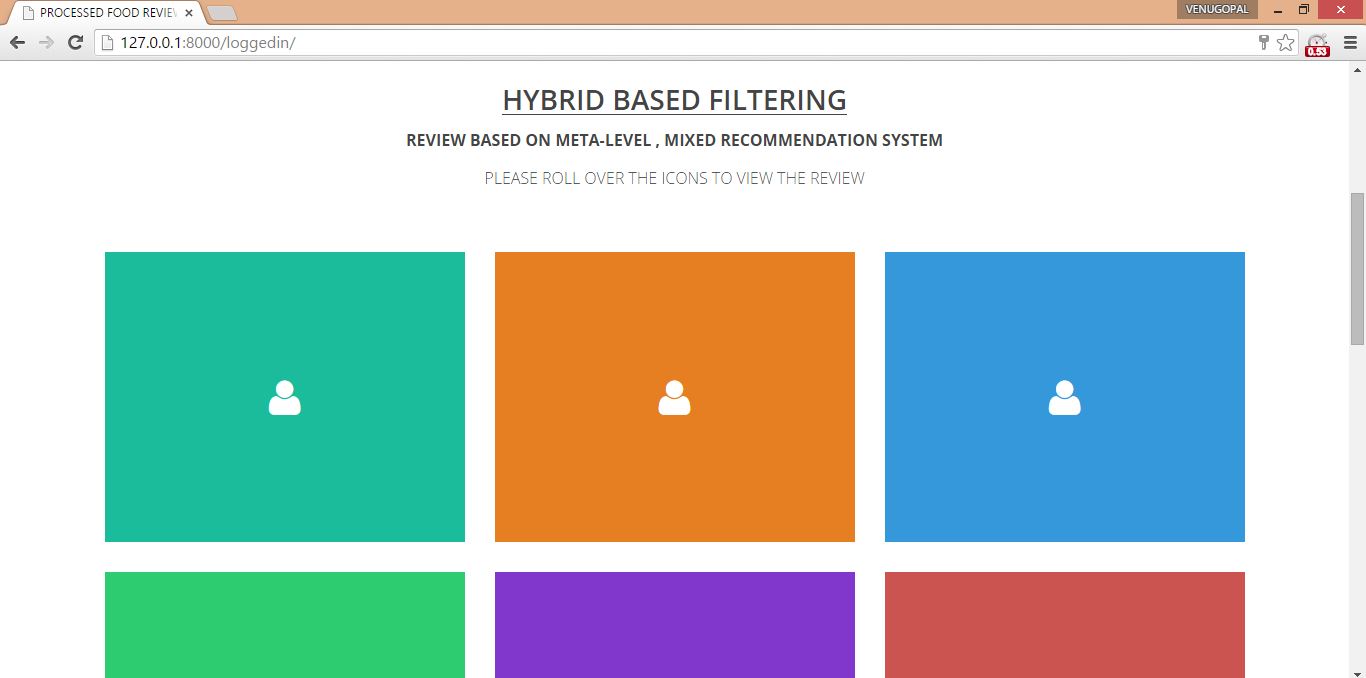
**5.2 Performance Analysis**

****

The performance analysis of the content based filtering is given on the top right corner of the page. This module apparently takes, 0.47second to display the result.



The performance analysis of the collaborative filtering is given on the top right corner. This module takes only 0.28 seconds to display the result. This shows that it is much more efficient than the previous module, i,e.., content based filtering.

****

This is the performance analysis of the third main module that is hybrid filtering system. This module takes, 0.53 seconds. Even though it’s a bit higher than the collaborative filtering, we have found that the efficiency of prediction is much higher in the hybrid recommendation system.

**CHAPTER 6**

**CONCLUSIONS AND FUTURE ENHANCEMENTS**

We have successfully created processed food review recommendation system by combining content based, collaborative and hybrid filtering. In content based, the results are displayed depending on the pre-defined score set by the user. In collaborative filtering, the items are recommended based on the user preferences of the active user. We have developed a hybrid algorithm which is a combination of both content based and collaborative filtering system. We used a meta level and mixed approach to develop this algorithm. To a certain extend through the implementation of hybrid recommendation system, we were able to overcome few of the shortcomings of content based and collaborative filtering techniques. During the performance analysis , it was found that the collaborative filtering technique , showed less time, but on the whole, accuracy of predictions were high for the hybrid filtering technique. In future, to further increase the efficiency of our project, we are planning to implement a cosine similarity matrix to find the neighbor’s of a particular user and to increase the efficiency of prediction.

**REFERENCES**

**Journal Paper**

[1] G. Liu, K. Y. Lee, and H. F. Jordan, "TDM and TWDM de Bruijn networks and shufflenets for optical communications," *IEEE Trans. Comp.*, vol. 46, pp. 695-701, June 1997.

**Books**

[2] Recommender systems handbook By Franceso Ricci. Lior Rokach. Bracha.shapira. Paul.B.kantor.

**Papers presented at conferences (unpublished)**

[3] H. A. Nimr, "Defuzzification of the outputs of fuzzy controllers," presented at 5th International Conference on Fuzzy Systems, Cairo, Egypt, 1996.

**Thesis or dissertation**

[4] H. Zhang, "Delay-insensitive networks," M.S. thesis, University of Waterloo, Waterloo, ON, Canada, 1997.

[5] M. W. Dixon, "Application of neural networks to solve the routing problem in communication networks," Ph.D. dissertation, Murdoch University, Murdoch, WA, Australia, 1999.

**APPENDIX -I**

**CODE**

from django.shortcuts import render

from django.shortcuts import render\_to\_response

from django.template import RequestContext

from django.contrib import auth

from django.contrib.auth.decorators import login\_required

from django.core.context\_processors import csrf

from django.http import HttpResponse,HttpResponseRedirect

from .forms import MyRegistrationForm

from .models import logini

import uuid

import os

from django.contrib.auth.forms import UserCreationForm

def home(request):

c={}

c.update(csrf(request))

return render\_to\_response('login.html',c)

def readfi():

x=os.path.join(os.path.dirname(os.path.dirname(\_\_file\_\_)),'recom\\test.txt')

f=open(x,'r')

pre='';

num\_lines = sum(1 for line in open(x))

y=num\_lines/8

dbscore=request.user.logi.scoreop

for l in 9:

ran=random.randit(0,y)

p=(ran\*8)+1

s=linecache.getline(x,p+4)

a,b=s.split(';')

if ( ( dbscore == '0' and (( b == ' 1.0' ) or ( b == ' 2.0' ) ) )or

( dbscore == '1' and (( b == ' 3.0' ) or ( b == ' 4.0' ) ) )or

( dbscore == '2' and ( b ==' 5.0') ) ):

s1=linecache.getline(x,p)

a1,b1=s1.split(';')

s2=linecache.getline(x,p+1)

a2,b2=s2.split(';')

s3=linecache.getline(x,p+2)

a3,b3=s3.split(';')

s4=linecache.getline(x,p+3)

a4,b4=s4.split(';')

pre += "<div class='col-md-4 col-sm-6'>" + \

"<div class='service-item' id='service-1'>" + \

"<div class='service-icon'>" + \

"<i class='fa fa-pencil-square-o'></i>" + \

"</div> <!-- /.service-icon -->" + \

"<div class='service-content'>" + \

" <div class='inner-service'>" + \

"Product Id :" + b1 + \

"Reviewer Id :" + b2 + \

"Reviewer Name:"+ b3 + \

"Helpfulness : "+ b4 + \

"Score :"+ b + \

"</div>" + \

"</div> <!-- /.service-content -->" + \

"</div> <!-- /#service-1 --> "+ \

"</div> <!-- /.col-md-4 -->"

else:

while ( (b==' 1.0')or(b==' 2.0') ):

ran=random.randit(0,y)

p=(ran\*8)+1

s=linecache.getline(x,p+4)

a,b=s.split(';')

if ( ( dbscore.score == '0' and (( b == ' 1.0' ) or ( b == ' 2.0' ) ) )or

( dbscore.score == '1' and (( b == ' 3.0' ) or ( b == ' 4.0' ) ) )or

( dbscore.score == '2' and ( b ==' 5.0') ) ):

s1=linecache.getline(x,p)

a1,b1=s1.split(';')

s2=linecache.getline(x,p+1)

a2,b2=s2.split(';')

s3=linecache.getline(x,p+2)

a3,b3=s3.split(';')

s4=linecache.getline(x,p+3)

a4,b4=s4.split(';')

pre += "<div class='col-md-4 col-sm-6'>" + \

"<div class='service-item' id='service-1'>" + \

"<div class='service-icon'>" + \

"<i class='fa fa-pencil-square-o'></i>" + \

"</div> <!-- /.service-icon -->" + \

"<div class='service-content'>" + \

"<div class='inner-service'>" + \

"Product Id :" + b1 + \

"Reviewer Id :" + b2 + \

"Reviewer Name:"+ b3 + \

"Helpfulness : "+ b4 + \

"Score :"+ b + \

"</div>" + \

"</div> <!-- /.service-content -->" + \

"</div> <!-- /#service-1 --> "+ \

"</div> <!-- /.col-md-4 -->"

return pre

def log\_in(request):

context = RequestContext(request)

if request.method == 'POST':

username = request.POST.get('username','')

password = request.POST.get('password','')

user = auth.authenticate(username=username, password=password)

if user is not None:

if user.is\_active:

auth.login(request,user)

return HttpResponseRedirect('/loggedin/')

else:

return HttpResponse("Inactive user.")

else:

return HttpResponseRedirect('error.html')

return render\_to\_response('error.html')

def logouti(request):

auth.logout(request)

return HttpResponseRedirect(recom.views.home)

def create\_acc(request):

if request.method == 'POST':

form = MyRegistrationForm(request.POST)

if form.is\_valid():

form.save()

return HttpResponseRedirect('/regsuccess/')

args = {}

args.update(csrf(request))

args['form'] = MyRegistrationForm()

return render\_to\_response('signup.html',args)

def regsuccess(request):

return render\_to\_response('regsuccess.html');

def errori(request):

return render\_to\_response('error.html');

@login\_required

def loggedin(request):

return render\_to\_response('in.html',

{'full\_name',request.user.username,

'post\_con',readfi(),

'u\_id',request.user.logi.userid})

"""

'u\_id',request.user.userid,

def sign\_up(request):

if request.method=='POST':

form=flogin(request.POST)

if form.is\_valid():

username = request.POST.get('user-name','')

emailid = request.POST.get('user-mail','')

passwd = request.POST.get('user-pwd','')

scoreop = request.POST.get('rang','')

loginip=logini(userid=my\_random\_string(14),username=username,emailid=emailid,passwd=passwd,scoreop=scoreop)

loginip.save()

return HttpResponseRedirect('/')

else:

form=flogin()

return render(request,'recome/index.html',{

'form':form,

}, RequestContext(request))

def loggedin(request):

return render\_to\_response('in.html',

{'full\_name',request.user.username,

'u\_id',request.user.userid,

'post\_con',readfi()})

def auth\_view(request):

if request.method=='POST':

form=flogi(request.POST)

if form.is\_valid():

uname=request.POST.get('user-email','')

pwd=request.POST.get('user-pw','')

user=logini.object.raw('Select \* from recom\_logini where emailid="'+uname+'" && passwd="'+pwd+'"')

else:

form=flogi();

if user is not None:

return render\_to\_response('in.html',

{'form',form,

'post\_con',readfi()}, RequestContext(request,{}))

else:

return render\_to\_response('index.html')

def loggedout(request):

return render\_to\_response('index.html')

def loginvalid(request):

return render\_to\_response('index.html')

"""