**INTERNSHIP REPORT**

**RESTURANT RECOMMENDATION SYSTEM**

Submitted in partial fulfillment of the requirements for the award of theDegree

of

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

Submitted by

**A. LALITHA (20KN1A0505)**

**M. RISHITHA (20KN1A0563) CH. BHAVYA SRI (20KN1AO531)**

**Under the guidance of Ms. Vindhyavasini**

**&**

### Under the Guidance of Mr. O. Shiv Bhagwan, M. Tech.

**Assoc. Professor**

### Department of Computer Science and Engineering RAMACHANDRA COLLEGE OF ENGINEERING

(Approved by AICTE, Affiliated to JNTUK, Kakinada)Accredited

by NBA, NAAC A+

NH-16 Bypass, Vatluru (V), Eluru -534007, W.G. Dist., A.P

### 2019 – 2023

|  |  |
| --- | --- |
|  | C:\Users\KAMESH\Downloads\IMG-20210706-WA0006.jpg |

**RAMACHANDRA COLLEGE OF ENGINEERING**

(Approved by AICTE, Affiliated to JNTUK, Kakinada) Accredited by NBA, NAAC A+

NH-16 Bypass, Vatluru (V), Eluru -534007, W.G. Dist., A.P

**DEPARTMENT OF COMPUTER SCIENCE& ENGINEERING**

## CERTIFICATE

This is to certify that the “**Internship report”** submitted by A. LALITHA **(20KN1A0505),**

M. RISHITHA (20KN1A0563) and CH. BHAVYA (20KN1A0531) is work done by his/her and submitted during YEARS academic year, in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING,** at **BLACKBUCK ENGINEERS PVT LTD.**



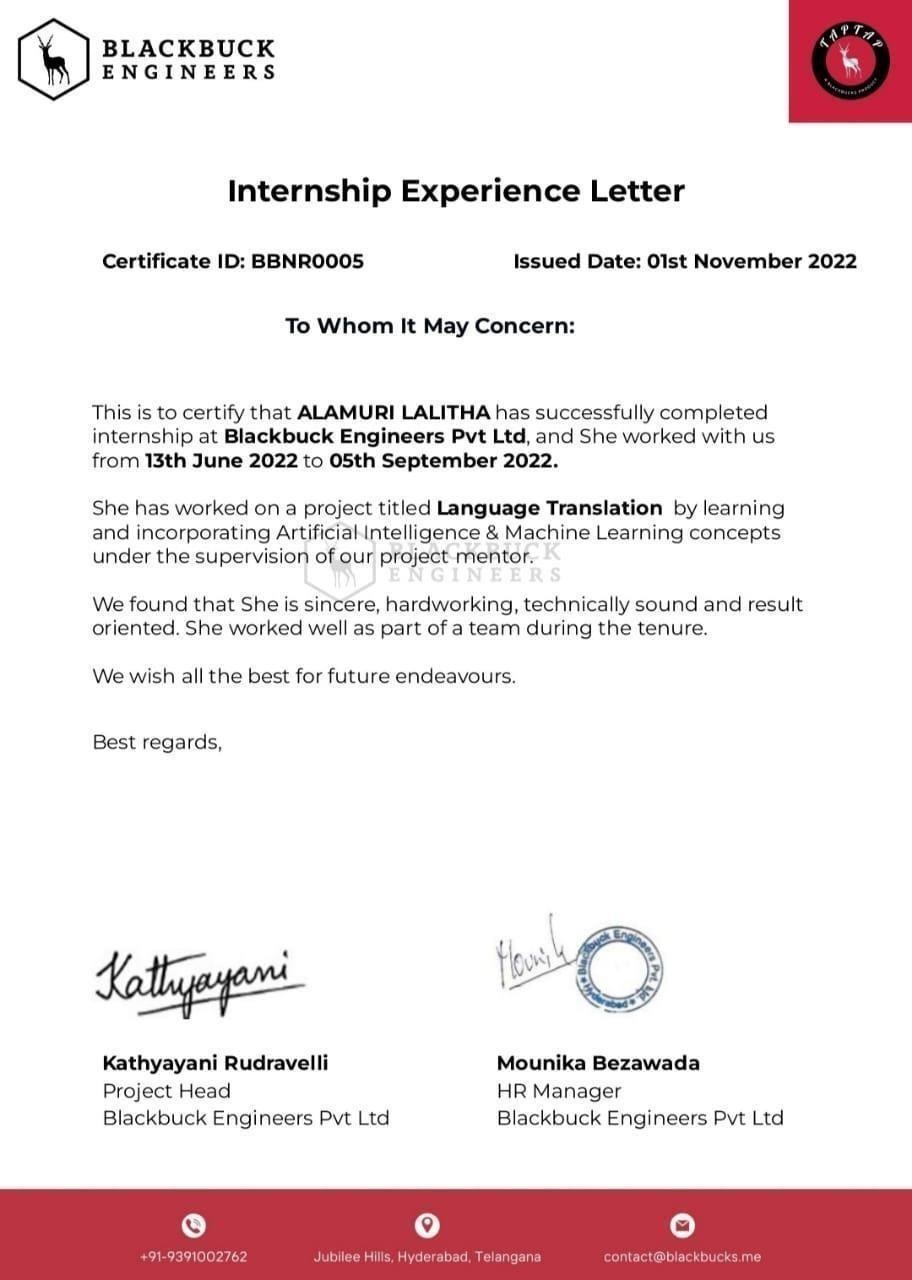
**INTERNSHIP COORDINATOR Head of the Department**

(R KATHYAYANI) Dr. P M Prasuna

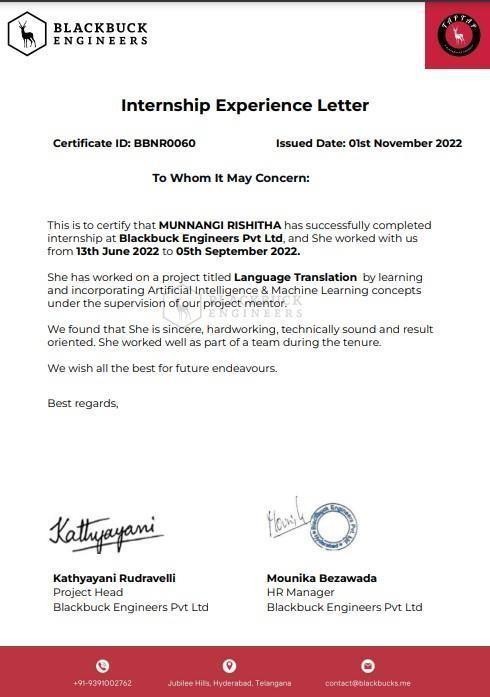
Associate Professor Professor & HOD,CSE

**EXTERNAL EXAMINER**

## CERTIFICATE OF INTERNSHIP



**CERTIFICATE OF INTERNSHIP**



## CERTIFICATE OF INTERNSHIP



### ACKNOWLEDGEMENT

We wish to take this opportunity to express our deep gratitude to all the people who have extended their cooperation in various ways during our project work. It is our pleasure and responsibility to acknowledge the help of all those individuals.

We sincerely thank our guide **Mr. O. Shiv Bhagwan,** Professor in the Departmentof CSE for helping us in successful completion of our project under his supervision.

We are very grateful to **Dr. P M Prasuna**, Head of the Department, Department of Computer Science & Engineering for her assistance and encouragement in all respects in carrying throughout our project work.

We express our deepest gratitude to **Dr. V Srinivasa Rao**, Principal, Ramachandra College of Engineering, and Eluru for his valuable suggestions during preparation of draftin our document.

We thank the project coordinators Dr Das and Dr A. Daveedu Raju, Professor, Department of CSE, for their valuable guidance and support throughout the development of this project.

We express our deepest gratitude to the management of Ramachandra College of engineering, Eluru and all the faculty members for their support and encouragement.

We sincerely thank Blackbuck Engineers and industry mentors for their valuable advice, suggestions which played a vital role in carrying out this project work

Finally, we thank one and all who directly or indirectly helped us to complete our project work successfully.

**A LALITHA(20KN1A0505) M RISHITHA (20KN1A0563)**

**CH BHAVYA SRI (20KN1A0531**

# ABSTRACT

The recommendation systems are the most effective ways for humans to make decisions based on the reviews of other people. The opinions, reviews and interests among the likeminded people are the best metrics that are used in the recommendation systems. One of the major needs of the society lies in culinary or just for the purpose of relaxation or entertainment, people tend to go to restaurants. However, people always prefer a restaurant with good reputation or a restaurant that fits their requirements/ tastes. Hence, we implement a technique that helps people choose the restaurant that have the facilities that they personally require, and to provide them with this information, the dataset of restaurants’ ratings, its specialty, reviews, the menu, ambience etc., are considered and when people use this recommendation system, they could use simple keywords and get the filtered results that satisfy their conditions. Thus, the restaurant recommendation system - simply a list of restaurants sorted on the basis of the customers’ ratings and when given the few interests of the users, the system brings up the filtered search results based on the Maximum Likelihood using these given attributes. Keywords: Recommendation systems, ratings, reviews, Maximum Likelihood.

The buying behavior of the consumer is affected by the suggestions given to the items. Recommendations can be made in the form of a review or ranking given to a specific product. Calories consumed by people contains carbohydrates, fats, proteins, minerals and vitamins, and any malnutrition causes severe health problems. In this paper, we propose a recommendation system which is trained on the basis of the recommendations received by the customer who has already used the product. Software recommends the product to the customer on the basis of the experience of the consumer using the same product. Each person has his or her own eating patterns, based on the preferences and dislikes of the user, indicating that personalized diet is important to sustain the success and health of the user.

**INDEX**

# CONTENTS

#### Introduction 1

#### Analysis 2

#### [Software requirements specifications 4](#_bookmark0)

#### Technologies 6

#### [PYTHON](#_bookmark1)

#### JUPYTER NOTEBOOK NUMPY TENSORFLOW KERAS

#### Coding 10

#### Screenshots 15

#### Conclusion 19

#### Bibliography 21

### Learning Objectives/Internship Objectives

* + Internships are generally thought of to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit fromTraining Internships in order to receive real world experience and develop their skills.
  + An objective for this position should emphasize the skills you already possess in the area and your interest in learning more
  + Internships are utilized in a number of different career fields, including architecture, engineering, healthcare, economics, advertising and many more.
  + Some internship is used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.
  + Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

**WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES**

|  |  |  |  |
| --- | --- | --- | --- |
| **1st WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 1.04.2023 | Saturday | Onboarding the Students & Introduce the Topic |
| 3.04.2023 | Monday | Session-1 Basics on Python |
| 4.04.2023 | Tuesday | Session-2 MI Supervised and Unsupervised Algorithm |
| 5.04.2023 | Wednesday | Session-3 Deep Learning |
| 6.04.2023 | Thursday | Session-4 Project Preprocessing |
| 7.04.2023 | Friday | Session-5 Model Training |
| 8.04.2023 | Saturday | Session-6 Model Project Development & Process Explanation |

|  |  |  |  |
| --- | --- | --- | --- |
| **2nd WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 10.04.2023 | Monday | Abstract Building |
| 11.04.2023 | Tuesday | Abstract Building |
| 12.04.2023 | Wednesday | Abstract Submission |
| 13.04.2023 | Thursday | Abstract Submission |
| 14.04.2023 | Friday | PPT Preparation |
| 15.04.2023 | Saturday | PPT Preparation |
| 16.04.2023 | Sunday | PPT Preparation |

|  |  |  |  |
| --- | --- | --- | --- |
| **3rd WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 17.04.2023 | Monday | PPT Preparation |
| 18.04.2023 | Tuesday | Mid Review |
| 19.04.2023 | Wednesday | Mid Review |
| 20.04.2023 | Thursday | Mid Review |
| 21.04.2023 | Friday | Mid Review |
| 22.04.2023 | Saturday | Coding and Documentation |
| 23.04.2023 | Sunday | Coding and Documentation |

|  |  |  |  |
| --- | --- | --- | --- |
| **4th WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 24.04.2023 | Monday | Coding and Documentation |
| 25.04.2023 | Tuesday | Final Review |
| 26.04.2023 | Wednesday | Final Review |
| 27.04.2023 | Thursday | Final Review |
| 28.04.2023 | Friday | Final Review |
| 29.04.2023 | Saturday | Final Review |
| 30.04.2023 | Sunday | Final Review |

**CHAPTER – 1 INRODUCTION**

# INTRODUCTION

Nowadays, the fact that information in the digital world is excessively large, complex, and dynamic often bores users. The main solution to overcome this problem is to guide users with recommendation systems to discover products or services in a personalized way. Recommendation Systems is the general term for systems that make recommendations for users based on their ratings, reviews, feedback, and likes, taking into account their search history. For example, the leading sites and apps in the digital world, such as Amazon, Netflix, Spotify, Apple, also provide recommendations to their users using their recommendation systems. While a high success rate in these recommendation systems increases user satisfaction, It also significantly increases the revenue generated by the applications because if they are shown the products they need at the right time and are increasingly encouraged to buy, the potential revenue per user will increases. The reason why we have turned to restaurant recommendation systems in this study is that online food applications are at the forefront of applications that have gained prominence in today's world and have become an indispensable part of our lives due to pandemic conditions. The problems with using these applications, the superficial customization of customers when ordering, and the fact that reviews that are important to customers are not transparently reflected were our motivation for this article. Recommender systems typically use models such as

1. Content-based Filtering Methods;
2. Collaborative Filtering Methods;
3. Hybrid Recommender Systems.

The goal of content-based recommender systems is to create a common profile for each user and item. This created profile is considered as an example of users. The specific details of users and products are important. This model learns from the user's preferences and finds items that are similar to the user's preferences then recommends them. Various methods such as the TF-IDF method, element vector, user vector, cosine similarity are used to calculate similarities between items. The purpose of the collaborative filtering model is to create a model based on user's past behavior and use it to predict their future behavior. The basis of this method is the logic that users with similar profiles can make

similar preferences. For example, if user X has the same opinion as user Y on a topic, they may both prefer a restaurant with the same type of cuisine. In the collaborative filtering method, the Pearson correlation coefficient, the Spearman correlation coefficient, the vector similarity, and the Cosine Similarity method can be used in calculating the similarities to build the user-item model. Hybrid recommendation systems, on the other hand, are the method that emerges from the need by combining content-based and collaborative recommender systems. In some studies in the literature, it has been found that the performance of this method is more accurate than the other methods under consideration.

[2] One of the most used applications of this hybrid system is Spotify. There are several key factors that influence the performance of recommender systems:

1. The "cold start" problem - insufficient user information and the recommendation system does not know user preferences
2. Limited recommendations - Recommendation systems are mostly limited to implicit user feedback.
3. Undecided users, manipulative ratings, confidentiality agreements to protect the user's personal data.
4. Incorrect selection of similarity algorithms. In recent years, deep learning, one of the sub-branches of artificial intelligence, which is a machine learning method, has become very popular. One of the ways to use these deep neural networks in recommender systems is to model the similarity relation. [3] but this method has not been used in studies in the field of restaurant recommendation systems. For this reason, this article investigated the performance of the Neural Collaborative Filtering Method in restaurant recommendation systems. In particular, we will explain in detail the machine learning algorithms (matrix factorization, multi-layer perceptron, hybrid algorithm combining matrix factorization, and multilayer perceptron NeuMF). Among our goals in this study is to find the neural collaborative filtering algorithm that has the highest success rate in the experiments we will perform on the restaurant dataset. The contents of the article are as follows: In the second section, studies on restaurant recommendation systems are reviewed; in the third section, the architectural design and algorithms of the neural collaborative filtering method are explained. Then, in the last section, the dataset, experiment, and results are explained.

similarities to build the user-item model. Hybrid recommendation systems, on the other hand, are the method that emerges from the need by combining content-based and collaborative recommender systems. In some studies in the literature, it has been found that the performance of this method is more accurate than the other methods under consideration. [2] One of the most used applications of this hybrid system is Spotify. There are several key factors that influence the performance of recommender systems:

* 1. The "cold start" problem - insufficient user information and the recommendation system does not know user preferences
  2. Limited recommendations - Recommendation systems are mostly limited to implicit user feedback.
  3. Undecided users, manipulative ratings, confidentiality agreements to protect the user's personal data.
  4. Incorrect selection of similarity algorithms. In recent years, deep learning, one of the sub- branches of artificial intelligence, which is a machine learning method, has become very popular. One of the ways to use these deep neural networks in recommender systems is to model the similarity relation. [3] but this method has not been used in studies in the field of restaurant recommendation systems. For this reason, this article investigated the performance of the Neural Collaborative Filtering Method in restaurant recommendation systems. In particular, we will explain in detail the machine learning algorithms (matrix factorization, multi-layer perceptron, hybrid algorithm combining matrix factorization, and multilayer perceptron NeuMF). Among our goals in this study is to find the neural collaborative filtering algorithm that has the highest success rate in the experiments we will perform on the restaurant dataset. The contents of the article are as follows: In the second section, studies on restaurant recommendation systems are reviewed; in the third section, the architectural design and algorithms of the neural collaborative filtering method are explained. Then, in the last section, the dataset, experiment, and results are explained.

**CHAPTER – 2 SYSTEM ANALYSIS**

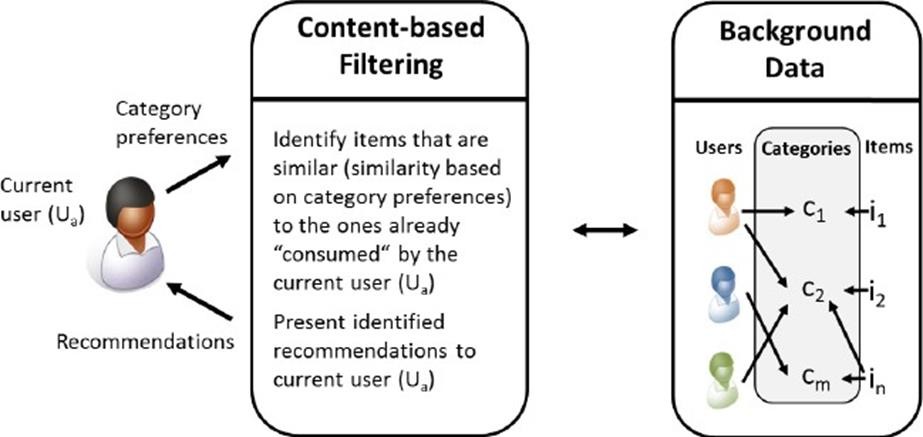
# SYSTEM ANALYSIS

#### EXISTING SYSTEM

Various methods are present for the development of restaurant recommendation system. Many of the existing systems and functioning are as follows. In this recommender system, they developed recommendation based on preferences of user. It was motivated by the observation that a user’s preference against an item is affected by different aspects discussed in reviews. They first explored the topic modelling to discover the hidden aspects from review text. Finally, they utilized regression models to detect the user-restaurant relationship. They described the restaurant recommendation system was very popular service whose accuracy and sophistication keeps increasing every day. They presented a personalized location-based restaurant recommendation system integrated in mobile technology. It was ubiquitously studied the user’s behavioral pattern of recommendation systems and proposed methods to rectify it. In this Research, they described the restaurant recommendation system with machine learning algorithms. In order to find a good machine-learning model, they have tried several collaborating filtering methods to predict ratings between restaurants and users. The methods they have implemented are Slope One, k-Nearest Neighbors algorithm, and multiclass SVM classification. Our evaluation shows that the multiclass SVM classification method outperforms the other methods. For rating prediction, they compare user-based and item-based collaborative filtering algorithms. Finally, architecture is given to support the building of a real-time recommendation service

#### PROPOSED SYSTEM

In this system, Content-based filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback.



the model should recommend items relevant to this user. lot do so, you must first pick a similarity metric (for example, dot product). then, you must set up the system to score each candidate item according to this similarity metric. Note that the recommendations are specific to this user, as the model did not use any information about other users



**CHAPTER – 3 SOFTWARE**

# SOFTWARE REQUIREMENTS SPECIFICATIONS

System configurations The software requirement specification can produce at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refinedby established a complete information description, a detailed functional description, a representation of system behavior, and indication of performance and design constrain, appropriate validate criteria, and other information pertinent to requirements.

Software Requirements:

* Operating system : Windows 7 Ultimate.
* Coding Language : python

Hardware Requirement:

* System : Pentium IV 2.4 GHz.
* Hard Disk : 1TB.
* Ram : 4GB.

# CHAPTER-4 TECHNOLOGIES

#### TECHNOLOGIES

Machine learning The goal of machine learning (ML) is to help a computer learn without being explicitly instructed to do so by means of mathematical models of data. Artificial intelligence (AI) is a subset of machine learning. Data is analysed using algorithms to identify patterns, which are then used to create predictive models. Like humans, machine learning becomes more accurate with more data and experience.

With machine learning, you can adapt to situations where data is constantly changing, the nature of the request or task is shifting, or coding a solution isn't feasible.

Algorithm for Random Forest

1. Select random samples from a given dataset and build multiple subsets.
2. Build a decision tree associated for every subset.
3. Every decision tree will output a prediction.
4. Take the average of these predicted values. v. The average value will be the final prediction.

#### PYTHON

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically- typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages.

#### JUPYTER NOTEBOOK

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole NRI INSTITUTE OF TECHNOLOGY computation process: developing, documenting, and executing code, as well as communicating the results. The Jupyter notebook combines two components: A web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output. Notebook documents: a representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects. Main features of the web application

* In-browser editing for code, with automatic syntax highlighting, indentation, and tab completion/introspection.
* The ability to execute code from the browser, with the results of computations attached to the code which generated them.
* Displaying the result of computation using rich media representations, such as HTML, LaTeX, PNG, SVG, etc. For example, publication-quality figures rendered by the matplotlib library, can be included inline.
* In-browser editing for rich text using the Markdown markup language, which can provide commentary for the code, is not limited to plain text.
* The ability to easily include mathematical notation within markdown cells using LaTeX, and rendered natively by MathJax.

NumPY

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:

* NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
* The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
* NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
* A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python- based software, just knowing how to use Python’s built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

#### Pandas

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

Recommender System Algorithms Most recommender systems take three basic approaches: collaborative filtering, content-based filtering and hybrid method

#### Collaborative Filtering (CF):

Items are recommended to users based on the past ratings of all users collectively.

#### Content-based recommending:

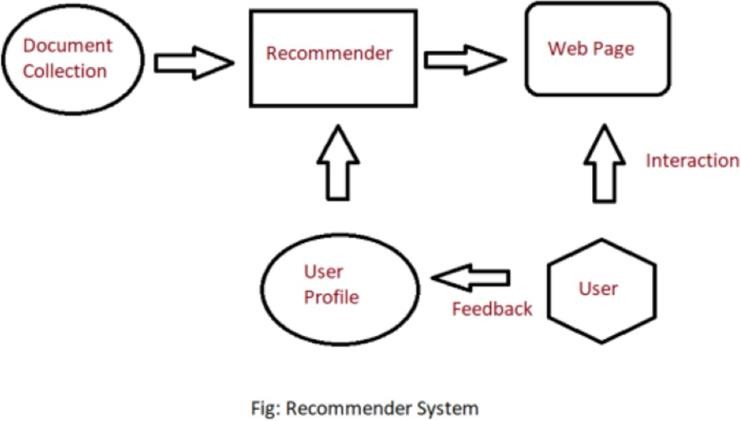
Items recommended to users are those that are similar in content to items the user has liked in the past, or matched to attributes of the user.

#### Hybrid approaches:

These methods combine both collaborative and content−based approaches.

#### Collaborative Filtering

In a collaborative filtering system, recommendation are made based on a model of prior user behavior. The model can either be constructed solely from a single user’s behavior or from the behavior of other users who have similar traits. Usually the latter approach is more 2 effective. When it takes other users’ behavior into consideration, collaborative filtering uses group knowledge to form a recommendation based on users. In essence, recommendations are based on an automatic collaboration of multiple users and filtered on those who exhibit similar preferences or behaviors. For example, suppose you’re building a website to recommend movies. By using the information from many users who have rated different movies, you can group those users based on their preferences. For example, you can group together users who gave similar rating to the same movies. From this information, you identify the highest rated movies that are rated by that group. Then, for a particular user in the group, you can recommend the highest rated movies that he or she has never rated or watched before.



#### Content-based recommendation

Content-based approach is based on a user’s behavior. For example, this approach might use historical information, such the ratings given by the user to the movies he or she previously watched. If a user gives high ratings to most of the action movies he watches, then content

based filtering can use this history to identify and recommend movies with similar content (action). This content can be manually defined or automatically extracted based on other similarity methods.

#### Hybrid

Hybrid approaches combine collaborative and content-based filtering and help to increase the efficiency (and complexity) of recommendation systems. Incorporating the results of collaborative and content-based filtering creates the potential for a more accurate recommendation. The hybrid approach could also be used to address collaborative filtering that starts with sparse data− known as cold start− by enabling the results to be weighted initially toward content-based filtering, then shifting the weight toward collaborative filtering as the available user dataset matures.

In this system, we develop a restaurant recommendation system using the Matrix Factorization or Latent Factor Collaborative Filtering Optimization. This system recommends restaurants for users based on their preferences such as beautiful ambience, good food, tasteful desserts and soon. This system provides personalized restaurant recommendations to users.

#### Algorithm Used

Recommender Systems or Recommendation Systems are straightforward algorithms that aim to provide the foremost relevant and correct items (products, movies, events,articles) to the user. There are two types of Recommendation Systems:

#### Content Based Filtering

Here the recommendations are done to the user based on the previous items that are highly rated by the user himself. But these don’t work efficiently for large data. A Content-Based Recommender works by the data that we take from the user, either explicitly (rating) or implicitly (clicking on a link). By the data we create a user profile, which is then used to suggest to the user, as the user provides more input or

take more actions on the recommendation, the engine becomes more accurate.

#### User Profile:

In the User Profile, we create vectors that describe the user’s preference. In the creation of a user profile, we use the utility matrix which describes the relationship between user and item. With this information, the best estimate we can make regarding which item user likes, is some aggregation of the profiles of those items.

#### Item Profile:

In Content-Based Recommender, we must build a profile for each item, which will represent the important characteristics of that item. For example, if we make a movie as an item then its actors, director, release year and genre are the most significant features of the movie. We can also add its rating from the IMDB (Internet Movie Database) in the Item Profile.

#### Utility Matrix:

Utility Matrix signifies the user’s preference with certain items. In the data gathered from the user, we have to find some relation between the items which are liked by the user and those which are disliked, for this purpose we use the utility matrix. In it we assign a particular value to each user-item pair, this value is known as the degree of preference. Then we draw a matrix of a user with the respective items to identify their preference relationship

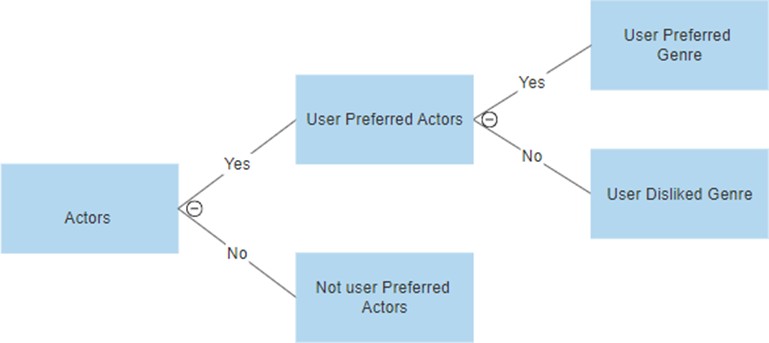
Recommending Items to User Based on Content:

* Method 1:

We can use the cosine distance between the vectors of the item and the user to determine its preference to the user. For explaining this, let us consider an example: We observe that the vector for a user will have a positive number for actors that tend to appear in movies the user likes and negative numbers for actors user doesn’t like, Consider a movie with actors which user likes and only a few actors which user doesn’t like, then the cosine angle between the user’s and movie’s vectors will be a large positive fraction. Thus, the angle will be close to 0, therefore a small cosine distance between the vectors. It represents that the user tends to like the movie, if the cosine distance is large, then we tend to avoid the item from the recommendation.

Method 2:

We can use a classification approach in the recommendation systems too, like we can use the Decision Tree for finding out whether a user wants to watch a movie or not, like at each level we can apply a certain condition to refine our recommendation. For example:



Advantages

* the model doesn't need any data about other users, since the recommendations are specific to this user. this makes it easier to scale to a large number of users.
* the model can capture the specific interests of a user, and can recommend niche items that very few other users are interested in.
  1. Collaborative Based Filtering

Here the recommendations are based on the idea that people who share same interest in certain kind of items will also share the same interest in some other kind of item

**CHAPTER-5 CODING**

#### CODING

1. **Loading the dataset:** Load the data and import the libraries.

#### Data Cleaning:

Deleting redundant columns. Renaming the columns.

Dropping duplicates. Cleaning individual columns.

Remove the NaN values from the dataset

#### Some Transformations

1. **Text Preprocessing**

Cleaning unnecessary words in the reviews Removing links and other unncessary items Removing Symbols

#### Recommendation System

Importing Libraries

*#Importing Libraries* **import** numpy **as** np **import** pandas **as** pd **import** seaborn **as** sb

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**from** sklearn.linear\_model **import** LogisticRegression **from** sklearn.linear\_model **import** LinearRegression **from** sklearn.model\_selection **import** train\_test\_split **from** sklearn.metrics **import** classification\_report **from** sklearn.metrics **import** confusion\_matrix

**from** sklearn.metrics **import** r2\_score **import** warnings warnings**.**filterwarnings('always') warnings**.**filterwarnings('ignore') **import** re

**from** nltk.corpus **import** stopwords

**from** sklearn.metrics.pairwise **import** linear\_kernel

**from** sklearn.feature\_extraction.text **import** CountVectorizer

**from** sklearn.feature\_extraction.text **import** TfidfVectorizer Loading the dataset

*#reading the dataset*

zomato\_real=pd.read\_csv("../input/zomato-bangalore-restaurants/zomato.csv") zomato\_real.head() # prints the first N rows of a DataFrame

zomato\_real.info()

Data Cleaning and Feature Engineering

#Deleting Unnnecessary Columns zomato=zomato\_real.drop(['url','dish\_liked','phone'],axis=1) #Dropping the column "dish\_liked", "phone", "url" and saving the new dataset as "zomato"

#Removing the Duplicates zomato.duplicated().sum() zomato.drop\_duplicates(inplace=True)

#Remove the NaN values from the dataset zomato.isnull().sum() zomato.dropna(how='any',inplace=True) zomato.info() #.info() function is used to get a concise summary of the dataframe

#Reading Column Names zomato.columns

#Changing the column names

zomato = zomato.rename(columns={'approx\_cost(for two people)':'cost','listed\_in(type)':'type', 'listed\_in(city)':'city'})

zomato.columns

#Some Transformations

zomato['cost'] = zomato['cost'].astype(str) #Changing the cost to string zomato['cost'] = zomato['cost'].apply(lambda x: x.replace(',','.')) #Using lambda function to replace ',' from cost zomato['cost'] = zomato['cost'].astype(float) # Changing the cost to Float

zomato.info()

#Reading Rate of dataset zomato['rate'].unique()

#Removing '/5' from Rates

zomato = zomato.loc[zomato.rate !='NEW']

zomato = zomato.loc[zomato.rate !='-'].reset\_index(drop=True) remove\_slash = lambda x: x.replace('/5', '') if type(x) == np.str else x zomato.rate = zomato.rate.apply(remove\_slash).str.strip().astype('float') zomato['rate'].head()

# Adjust the column names

zomato.name = zomato.name.apply(lambda x:x.title()) zomato.online\_order.replace(('Yes','No'),(True, False),inplace=True) zomato.book\_table.replace(('Yes','No'),(True, False),inplace=True) zomato.cost.unique()

zomato.head() zomato['city'].unique()

zomato**.**head()

*## Checking Null values*

zomato**.**isnull()**.**sum()

*## Computing Mean Rating*

restaurants **=** list(zomato['name']**.**unique()) zomato['Mean Rating'] **=** 0

**for** i **in** range(len(restaurants)):

zomato['Mean Rating'][zomato['name'] **==** restaurants[i]] **=** zomato['rate'][zomato['name'] **==** restaurants[i]]**.**mean() zomato**.**head() **from** sklearn.preprocessing **import** MinMaxScaler

scaler **=** MinMaxScaler(feature\_range **=** (1,5))

zomato[['Mean Rating']] **=** scaler**.**fit\_transform(zomato[['Mean Rating']])**.**round(2)

zomato**.**sample(3) zomato**.**head()

*## Text Preprocessing*

Some of the common text preprocessing / cleaning steps are:

Lower casing

Removal of Punctuations Removal of Stopwords Removal of URLs Spelling correction

*# 5 examples of these columns before text processing:*

zomato[['reviews\_list', 'cuisines']]**.**sample(5)

*## Lower Casing*

zomato["reviews\_list"] **=** zomato["reviews\_list"]**.**str**.**lower() zomato[['reviews\_list', 'cuisines']]**.**sample(5)

*## Removal of Puctuations*

**import** string

PUNCT\_TO\_REMOVE **=** string**.**punctuation

**def** remove\_punctuation(text):

"""custom function to remove the punctuation"""

**return** text**.**translate(str**.**maketrans('', '', PUNCT\_TO\_REMOVE))

zomato["reviews\_list"] **=** zomato["reviews\_list"]**.**apply(**lambda** text: remove\_punctuation(text)) zomato[['reviews\_list', 'cuisines']]**.**sample(5)

*## Removal of Stopwords*

**from** nltk.corpus **import** stopwords STOPWORDS **=** set(stopwords**.**words('english')) **def** remove\_stopwords(text):

"""custom function to remove the stopwords"""

**return** " "**.**join([word **for** word **in** str(text)**.**split() **if** word **not in**

STOPWORDS])

zomato["reviews\_list"] **=** zomato["reviews\_list"]**.**apply(**lambda** text: remove\_stopwords(text))

*## Removal of URLS*

**def** remove\_urls(text):

url\_pattern **=** re**.**compile(r'https?://\S+|www\.\S+')

**return** url\_pattern**.**sub(r'', text)

zomato["reviews\_list"] **=** zomato["reviews\_list"]**.**apply(**lambda** text: remove\_urls(text)) zomato[['reviews\_list', 'cuisines']]**.**sample(5)

*# RESTAURANT NAMES:*

restaurant\_names **=** list(zomato['name']**.**unique()) restaurant\_names

**def** get\_top\_words(column, top\_nu\_of\_words, nu\_of\_word):

vec **=** CountVectorizer(ngram\_range**=** nu\_of\_word, stop\_words**=**'english') bag\_of\_words **=**

vec**.**fit\_transform(column)

sum\_words **=** bag\_of\_words**.**sum(axis**=**0)

words\_freq **=** [(word, sum\_words[0, idx]) **for** word, idx **in**

vec**.**vocabulary\_**.**items()]

words\_freq **=**sorted(words\_freq, key **= lambda** x: x[1], reverse**=True**)

**return** words\_freq[:top\_nu\_of\_words] zomato**.**head() zomato**.**sample(5) zomato**.**shape zomato**.**columns

zomato**=**zomato**.**drop(['address','rest\_type', 'type', 'menu\_item', 'votes'],axis**=**1)

**import** pandas

*# Randomly sample 60% of your dataframe*

df\_percent **=** zomato**.**sample(frac**=**0.5)

df\_percent**.**shape

Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency (TF-IDF) vectors for each document. This will give you a matrix where each column represents a word in the overview vocabulary (all the words that appear in at least one document) and each column represents a restaurant, as before.

TF-IDF is the statistical method of evaluating the significance of a word in a given document. TF — Term frequency(tf) refers to how many times a given term appears in a document.

IDF — Inverse document frequency(idf) measures the weight of the word in the document, i.e if the word is common or rare in the entire document. The TF-IDF intuition follows that the terms that appear frequently in a document are less important than terms that rarely appear. Fortunately, scikit-learn gives you a built-in TfIdfVectorizer class that produces the TF-IDF matrix quite easily.

df\_percent**.**set\_index('name', inplace**=True**) indices **=** pd**.**Series(df\_percent**.**index)

*# Creating tf-idf matrix*

tfidf **=** TfidfVectorizer(analyzer**=**'word', ngram\_range**=**(1, 2), min\_df**=**0, stop\_words**=**'english')

tfidf\_matrix **=** tfidf**.**fit\_transform(df\_percent['reviews\_list']) cosine\_similarities **=** linear\_kernel(tfidf\_matrix, tfidf\_matrix) **def** recommend(name, cosine\_similarities **=** cosine\_similarities):

*# Create a list to put top 10 restaurants*

recommend\_restaurant **=** []

*# Find the index of the hotel entered*

idx **=** indices[indices **==** name]**.**index[0]

*# Find the restaurants with a similar cosine-sim value and order them from bigges number*

score\_series **=**

pd**.**Series(cosine\_similarities[idx])**.**sort\_values(ascending**=False**)

*# Extract top 30 restaurant indexes with a similar cosine-sim value*

top30\_indexes **=** list(score\_series**.**iloc[0:31]**.**index)

*# Names of the top 30 restaurants*

**for** each **in** top30\_indexes: recommend\_restaurant**.**append(list(df\_percent**.**index)[each])

*# Creating the new data set to show similar restaurants*

df\_new **=** pd**.**DataFrame(columns**=**['cuisines', 'Mean Rating', 'cost'])

*# Create the top 30 similar restaurants with some of their columns*

**for** each **in** recommend\_restaurant:

df\_new **=** df\_new**.**append(pd**.**DataFrame(df\_percent[['cuisines','Mean Rating', 'cost']][df\_percent**.**index **==**

each]**.**sample()))

*# Drop the same named restaurants and sort only the top 10 by the highest rating* df\_new **=** df\_new**.**drop\_duplicates(subset**=**['cuisines','Mean Rating', 'cost'], keep**=False**) df\_new **=** df\_new**.**sort\_values(by**=**'Mean Rating', ascending**=False**)**.**head(10)

print('TOP %s RESTAURANTS LIKE %s WITH SIMILAR REVIEWS: ' **%**

(str(len(df\_new)), name))

**return** df\_new

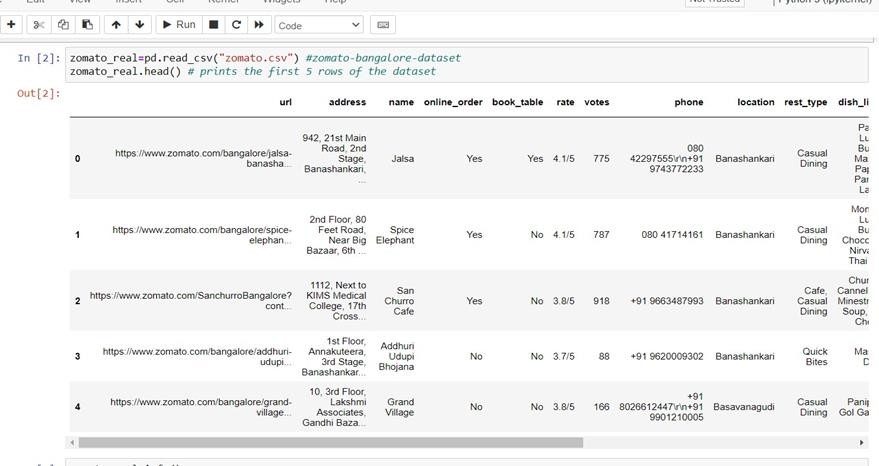
*# HERE IS A RANDOM RESTAURANT. LET'S SEE THE DETAILS ABOUT THIS RESTAURANT:*

df\_percent[df\_percent**.**index **==** 'Pai Vihar']**.**head() recommend('Pai Vihar')

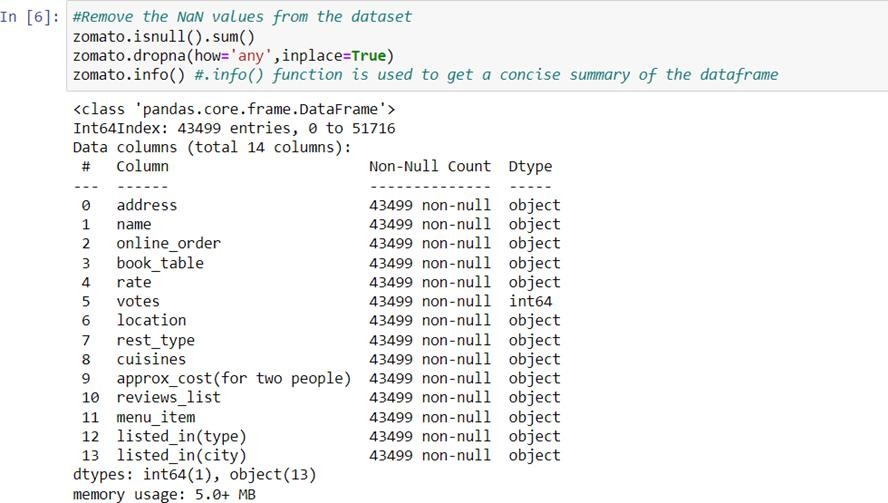
## CHAPTER-6 SCREEN SHOTS

#### SCREENSHOTS

**Prints first five rows:**



#### Remove null values:



**Computing mean rating**



**Top restaurants**



## CHAPTER -7 CONCLUSION

#### CONCLUSION

The main objective of the restaurant recommendation project is to let people check great restaurants through the website. They can get the restaurant information and price here, so that they do not call restaurants or check the yellow page for detail. They can find great restaurants by people’s ratings or recommendation to get suggestions. Those are from people’s experience in other way people try new restaurants from other other’s suggestion, the restaurant’s get more customers. It is an encouragement for great restaurants.

## CHAPTER-8 BIBILOGRAPHY

#### BIBLOGRAPHY

1. G. Adomavicius and A. Tuzhilin, \Towards the next generation of recommender systems:

A survey of the state-of-the-art and possible extensions," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734{749, June 2005.

1. D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, Recommender Systems: An Introduction. Cambridge University Press, September 2010.
2. P. Melville and V. Sindhwani, \Recommender systems," Encyclopedia of Machine Learning, 2010.
3. M. J. Pazzani and D. Billsus, \Content-based recommendation systems," The Adaptive Web, pp. 325{341, 2007.
4. K. Takano and K. F. Li, \An adaptive personalized recommender based on webbrowsing behaviour learning," IEEE Computer Society, pp. 654{660, 2009.
5. M. S. Pera and Y.-K. Ng, \A group recommender for movies based on content similarity and popularity," Information Processing and Management:
6. M. Balabanovic and Y. Shoham, \Content-based, collaborative recommendation," Communications of the ACM, vol. 40, no. 3, 1997.
7. Y.-L. Chen, L.-C. Cheng, and C.-N. Chuang, \A group recommendation system with consideration of interactions among group members," Expert Systems with Applica- tions, no. 34, pp. 2082{2090, 2008.
8. S. Amer-Yahia, S. B. Roy, A. Chawla, G. Das, and C. Yu, \Group recommendation: Semantics and e\_ciency," Proceedings of Very Large Data Base Endowment, vol. 2, pp. 754{756, August 2009.
9. L. Boratto and S. Carta, \State-of-the-art in group recommendation and new approaches

for automatic identi\_cation of groups," Information Retrieval and Mining in Distributed Environments, pp. 1{20, 2011.