Kunal Makwana (19BAI1011)
Prakriti Singh (19BAI1139)
Arjun Tandon (19BAI1146)

CSE1015
Machine Learning Essentials

Faculty: Prof. Rajalakshmi R

J-Component Review 3

Technical Report

Restaurant Recommendation System

Abstract

For our Machine Learning project, we have chosen Restaurant Recommendation System as the topic. Restaurant Recommendation System (RRS) is a type of recommender system that uses **content-based filtering**. This method only uses information about the description and attributes of items that users have previously consumed to model user preferences. By this we mean that in our system, unlike other systems used in food delivery apps such as Zomato and Swiggy, instead of the food being recommended first by

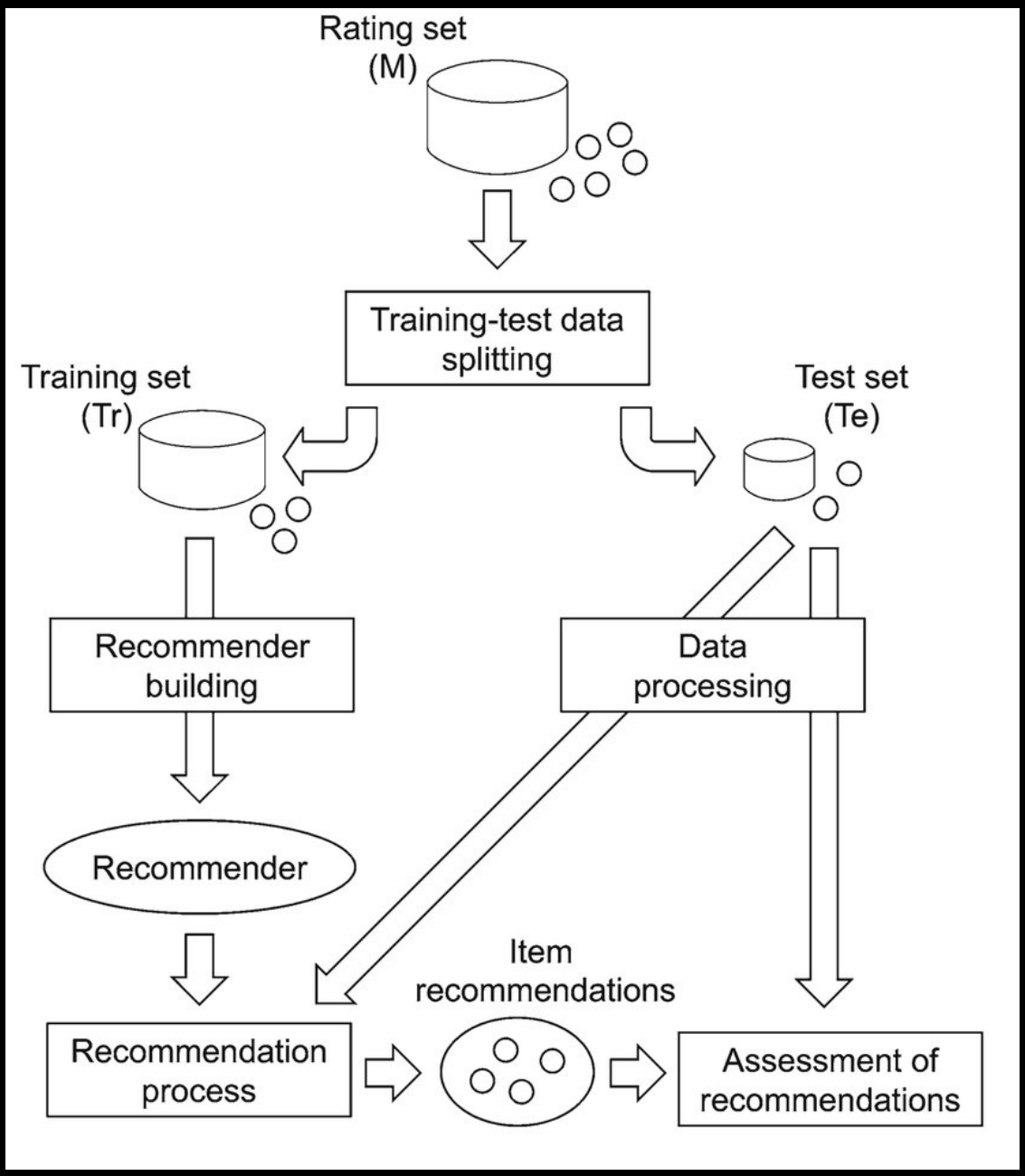
the distance of the restaurant from your location, we shall be recommending restaurants to visit based on the reviews received by those restaurants by other customers as well. These restaurants will be given priority on the basis of the user's previous orders and preferences. We shall create a content-based recommendation system where when the user enters the name of a restaurant, the RRS will look at reviews from other restaurants, and will recommend them to other restaurants with similar reviews and sort them from the top-rated in a descending order.

Introduction
Why a recommender system?

There is an extended class of applications that involve predicting user responses to a variety of options. Such a system is called a recommender system. The rapid growth in data collection has led to a new data-driven world. Data is used to create more efficient systems and there's where recommender system comes in. Recommender systems are intelligent systems which make suggestions about user items. Recommender system has become an important part of any entertainment or marketing website. As the recommender system has become so important it is a hot topic for any researcher. Recommender system is one application plugin that is being used by many vectors and online service providers to understand the needs of online users. Based on their need, the system is able to suggest them the best suitable product or match.

How does it work?

The recommender system is represented as an intelligent system, which identifies the user category based on user information analysis and user interest analysis. Once such information is obtained, in second stage, the analysis is performed to obtain the similarity group respective to available products and services. To



perform such kind of analysis there are some existing content based as well as collaborative recommender systems.

Related Works

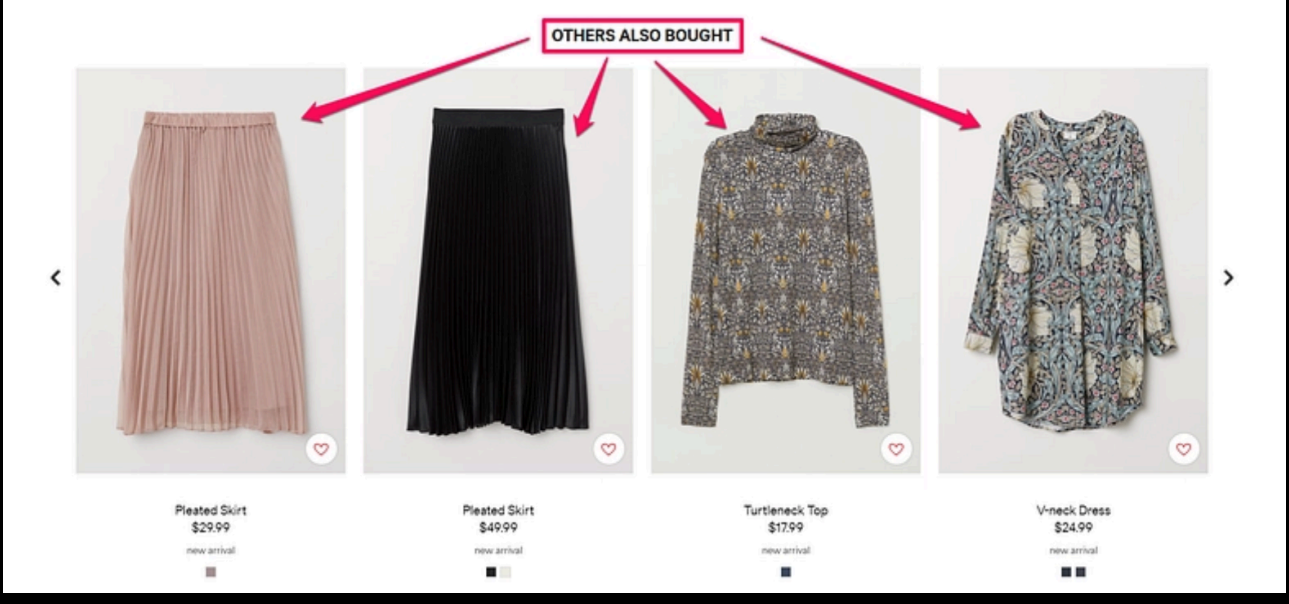
Netflix, YouTube, Tinder, and Amazon are all examples of recommender systems in use. The systems entice users with relevant suggestions based on the choices they make.

Recommender systems can also enhance experiences for:

- News Websites
- Computer Games
- Knowledge Bases
- Social Media Platforms
- Stock Trading Support Systems

Bottom line? If you want to provide users with targeted choices, recommender systems are the answer.

An example of a recommender system in e-commerce is in H&M's online shopping website. H&M served the following recommendations to users who clicked on "pleated skirt" as a potential buy:



Methodology

The requirements of the application were clear and waterfall model was used to develop the application.

The dataset we'll be using here consists of restaurants in Bangalore, India, collected from Zomato. We received this data from the website https://www.kaggle.com.

	address	name	online_order	book_table	rate
ww.zom angalo ari? yJzZSI NTg2OT 1NDc0I	942, 21st Main Road, 2nd Stage, Banashankari, Bangalore	Jalsa	Yes	Yes	4.1/5
ww.zom angalo ari? yJzZSI IjU4Nj ...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th Block, Kathriguppe, 3rd Stage, Banashankari, Bangalore	Spice Elephant	Yes	No	4.1/5
ww.zom anchur re? yJzZSI IjU4Nj zNzU0N kwLC..	1112, Next to KIMS Medical College, 17th Cross, 2nd Stage, Banashankari, Bangalore	San Churro Cafe	Yes	No	3.8/5
ww.zom angalo i- jana- ari? yJzZSI IjU4Nj	1st Floor, Annakuteera, 3rd Stage, Banashankari, Bangalore	Addhuri Udupi Bhojana	No	No	3.7/5

	+91 9743772233		
787	080 41714161	Banashankari	Casual Dining
918	+91 9663487993	Banashankari	Cafe, Casual Dining
88	+91 9620009302	Banashankari	Quick Bites

We use **item-item filtering** here which means if user A likes an item x, then, the items y and z which are similar to x in property, then y and z are recommended to the user. As a statement, it can be said, "Because you liked this, you may also like those".

The equation used here is:

$$R_{xu}=(\Sigma_{i=0}^{n}R_i) / n$$

Where R is the rating user u gives to the product x, and it is the average of the ratings u gave to products like x. Here also, we take a weighted average

$$R_{xu}=(\Sigma_{i=0}^{n}R_iW_i) / \Sigma_{i=0}^{n}W_i$$

Where the Weight is the similarity between the products.

The application is based on Flask framework. It uses python programming language in back-end and JavaScript in front-end. MySQL was used as DBMS for the application. The algorithms were implemented in python. JavaScript was used for validation. Similarly, MS Excel was used for data preprocessing and draw.io was used as case tool.

Experiments and Testing

Main python libraries that we shall be using are imported as follows:

```
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
```

The next step is 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)

#Changing the column names
zomato = zomato.rename(columns={'approx_cost(for two people)':'cost','listed_in(type)':'type', 'listed_in(city)':'city'})

#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)
#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')

# 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)

## 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()
```

	from sklearn.preprocessing import MinMaxScaler
	scaler = MinMaxScaler(feature_range = (1,5))
	zomato[['Mean Rating']] = scaler.fit_transform(zomato[['Mean Rating']]).round(2)

The next step is to perform some text preprocessing steps:-

## Lower Casing	
	zomato["reviews_list"] = zomato["reviews_list"].str.lower()
	## 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))
	## 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)
```

Output:-

TOP 10 RESTAURANTS LIKE Keto Kitchen WITH SIMILAR REVIEWS:

	cuisines	Mean Rating	cost
Banjara Melting Pot	North Indian, Chinese, Seafood	4.10	1.7
Brahmin Tiffins & Coffee	South Indian	4.10	100.0
Shake It Off	Beverages, Fast Food	3.96	600.0
Bowring Kulfi	Ice Cream, Desserts	3.84	150.0
Mc Donald'S	Burger, Fast Food	3.71	500.0
Rayalaseema Chefs	North Indian, Biryani, Andhra, Chinese	3.71	800.0
Mountain Spice	Chinese, Nepalese, Tibetan, Momos	3.71	400.0
Sulaimani Chai	Fast Food, Beverages	3.58	100.0
Punjab Mail	North Indian, Chinese, Beverages	3.58	650.0
Barbeque Delight	Arabian, Kerala, North Indian, Chinese, BBQ	3.48	800.0

# RESTAURANT NAMES:	
	restaurant_names = list(zomato['name'].unique())
	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=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)

References

1. Application of content-Based approach in research paper recommendation system for a digital library:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.673.6417&rep=rep1&type=pdf#page=49

2. Survey on collaborative **filtering**, **content-based filtering** and hybrid **recommendation system**:

https://www.academia.edu/download/59762468/10.1.1.695.642820190617-91457-z4s1rf.pdf

3. A Framework for Collaborative, Content-Based and Demographic Filtering:

https://link.springer.com/article/10.1023/A:1006544522159