Group 36 IDS Assignment

RESTAURANT RECOMMENDER SYSTEM

Creating a content based recommender system in which when the user types in a restaurant name, he/she gets the recommendations of the top 10 similar restaurants to the given input in sorted order with the highest rated on the top

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from warnings import filterwarnings
filterwarnings('ignore')
```

In [2]:
```
df_names = pd.read_csv('Restaurant name and related info.csv')
df_reviews = pd.read_csv('Restaurant_Review.csv')
```

In [3]:
```
df_names.head()
```

Out[3]:

	Name	Links	Cost	Collections	Cuisines	Timings
0	Beyond Flavours	https://www.zomato.com/hyderabad/beyond-flavou...	800	Food Hygiene Rated Restaurants in Hyderabad, C...	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)
1	Paradise	https://www.zomato.com/hyderabad/paradise-gach...	800	Hyderabad's Hottest	Biryani, North Indian, Chinese	11 AM to 11 PM
2	Flechazo	https://www.zomato.com/hyderabad/flechazo-gach...	1,300	Great Buffets, Hyderabad's Hottest	Asian, Mediterranean, North Indian, Desserts	11:30 AM to 4:30 PM, 6:30 PM to 11 PM
3	Shah Ghouse Hotel & Restaurant	https://www.zomato.com/hyderabad/shah-ghouse-h...	800	Late Night Restaurants	Biryani, North Indian, Chinese, Seafood, Bever...	12 Noon to 2 AM
4	Over The Moon Brew Company	https://www.zomato.com/hyderabad/over-the-moon...	1,200	Best Bars & Pubs, Food Hygiene Rated Restauran...	Asian, Continental, North Indian, Chinese, Med...	12noon to 11pm (Mon, Tue, Wed, Thu, Sun), 12no...

In [4]:
```
df_names.shape
```

Out[4]:
```
(105, 6)
```

In [5]:
```
df_reviews.head()
```

Out[5]:

	Restaurant	Reviewer	Review	Rating	Metadata	Time	Pictures
0	Beyond Flavours	Rusha Chakraborty	The ambience was good, food was quite good . h...	5	1 Review , 2 Followers	5/25/2019 15:54	0
1	Beyond Flavours	Anusha Tirumalaneedi	Ambience is too good for a pleasant evening. S...	5	3 Reviews , 2 Followers	5/25/2019 14:20	0
2	Beyond Flavours	Ashok Shekhawat	A must try.. great food great ambience. Thnx f...	5	2 Reviews , 3 Followers	5/24/2019 22:54	0
3	Beyond Flavours	Swapnil Sarkar	Soumen das and Arun was a great guy. Only beca...	5	1 Review , 1 Follower	5/24/2019 22:11	0
4	Beyond Flavours	Dileep	Food is good.we ordered Kodi drumsticks and ba...	5	3 Reviews , 2 Followers	5/24/2019 21:37	0

In [6]:
```
df_reviews.shape
```

Out[6]:
```
(10000, 7)
```

In [7]:
```
df_names.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 105 entries, 0 to 104
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Name         105 non-null    object
 1   Links        105 non-null    object
 2   Cost         105 non-null    object
 3   Collections  51 non-null     object
 4   Cuisines     105 non-null    object
 5   Timings      104 non-null    object
dtypes: object(6)
memory usage: 5.0+ KB
```

In [8]:
```
df_names.nunique()
```

Out[8]:
```
Name           105
Links          105
Cost            29
Collections     42
Cuisines        92
Timings         77
dtype: int64
```

In [9]:
```
df_reviews.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Restaurant  10000 non-null  object
 1   Reviewer    9962 non-null   object
 2   Review      9955 non-null   object
 3   Rating      9962 non-null   object
 4   Metadata    9962 non-null   object
 5   Time        9962 non-null   object
 6   Pictures    10000 non-null  int64 
dtypes: int64(1), object(6)
memory usage: 547.0+ KB
```

In [10]:
```
df_reviews.nunique()
```

Out[10]:
```
Restaurant     100
Reviewer      7446
Review        9364
Rating          10
Metadata      2477
Time          9782
Pictures        36
dtype: int64
```

We now merge the two datasets and inorder to do so we have to maintain same column names for Restaurant Name in both tables

In [11]:
```
# Renaming the restaurant name column with the same column name as in the other data set to merge the data sets
df_reviews = df_reviews.rename(columns={'Restaurant': 'Name'})

# Merging the two data sets
df = pd.merge(df_reviews, df_names, how='left', on='Name')
```

In [12]:
```
# Dropping the columns which are not of any significance
df.drop(['Reviewer', 'Time', 'Pictures', 'Links', 'Collections'], axis=1, inplace=True)
df.head()
```

Out[12]:

	Name	Review	Rating	Metadata	Cost	Cuisines	Timings
0	Beyond Flavours	The ambience was good, food was quite good . h...	5	1 Review , 2 Followers	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)
1	Beyond Flavours	Ambience is too good for a pleasant evening. S...	5	3 Reviews , 2 Followers	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)
2	Beyond Flavours	A must try.. great food great ambience. Thnx f...	5	2 Reviews , 3 Followers	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)
3	Beyond Flavours	Soumen das and Arun was a great guy. Only beca...	5	1 Review , 1 Follower	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)
4	Beyond Flavours	Food is good.we ordered Kodi drumsticks and ba...	5	3 Reviews , 2 Followers	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)

In [13]:
```
# Changing the data types of cost and rating columns
df['Cost'] = df['Cost'].str.replace(',', '').astype(int)
df['Rating'] = df['Rating'].str.replace('Like', '1').astype(float)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10000 entries, 0 to 9999
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      10000 non-null  object 
 1   Review    9955 non-null   object 
 2   Rating    9962 non-null   float64
 3   Metadata  9962 non-null   object 
 4   Cost      10000 non-null  int64  
 5   Cuisines  10000 non-null  object 
 6   Timings   9900 non-null   object 
dtypes: float64(1), int64(1), object(5)
memory usage: 625.0+ KB
```

Data Cleansing

In [14]:
```
print('No. of records:', len(df))
print('\nNo. of null values for each column:\n')
print(df.isnull().sum())
```

```
No. of records: 10000

No. of null values for each column:

Name          0
Review       45
Rating       38
Metadata     38
Cost          0
Cuisines      0
Timings     100
dtype: int64
```

As rating is important for recommendation we will not drop the null values

In [15]:
```
df['Name'][df['Rating'].isnull() == True].value_counts()
```

Out[15]:
```
American Wild Wings    23
Arena Eleven           15
Name: Name, dtype: int64
```

This indicates there are two restaurants having null rating values

In [16]:
```
print('Mean Rating of American Wild Wings: ', df['Rating'][df['Name'] == 'American Wild Wings'].mean())
print('Mean Rating of Arena Eleven: ', df['Rating'][df['Name'] == 'Arena Eleven'].mean())
print('Overall Mean Ratings: ', df['Rating'].mean())
```

```
Mean Rating of American Wild Wings:  3.9740259740259742
Mean Rating of Arena Eleven:  4.117647058823529
Overall Mean Ratings:  3.6007829753061635
```

We can impute these missing values in ratings feature by the mean rating value i.e, 4

In [17]:
```
df['Rating'].fillna(4, inplace=True)

# Changing NaN reviews by '-'
df['Review'] = df['Review'].fillna('-')
df.isnull().sum()
```

Out[17]:
```
Name          0
Review        0
Rating        0
Metadata     38
Cost          0
Cuisines      0
Timings     100
dtype: int64
```

Separating the Review and Follower columns into different columns

In [18]:
```
# Filling missing values:
df['Metadata'].fillna('0 Review , 0 Follower', inplace=True)
df['Metadata'].head(5)
```

Out[18]:
```
0     1 Review , 2 Followers
1    3 Reviews , 2 Followers
2    2 Reviews , 3 Followers
3      1 Review , 1 Follower
4    3 Reviews , 2 Followers
Name: Metadata, dtype: object
```

In [19]:
```
# Standardizing strings
df['Metadata'] = df['Metadata'].str.replace('Reviews', 'Review')
df['Metadata'] = df['Metadata'].str.replace('Followers', 'Follower')

df['Metadata'][df['Metadata'].str.endswith('w')] = df['Metadata'][df['Metadata'].str.endswith('w')] + ' , - Follower'
df['Metadata'].head(5)
```

Out[19]:
```
0    1 Review , 2 Follower
1    3 Review , 2 Follower
2    2 Review , 3 Follower
3    1 Review , 1 Follower
4    3 Review , 2 Follower
Name: Metadata, dtype: object
```

In [20]:
```
# Splitting into two columns
df[['Reviews', 'Followers']] = df['Metadata'].str.split(' , ', expand=True)
df[['Reviews', 'Followers']].head(5)
```

Out[20]:

	Reviews	Followers
0	1 Review	2 Follower
1	3 Review	2 Follower
2	2 Review	3 Follower
3	1 Review	1 Follower
4	3 Review	2 Follower

In [21]:
```
# Removing the words 'Review' and 'Follower' from the columns and storing only the count
df['Reviews'] = df['Reviews'].str.replace('Review', '')
df['Reviews'] = df['Reviews'].str.replace('Posts', '')
df['Reviews'] = df['Reviews'].str.replace('Post', '')

df['Followers'] = df['Followers'].str.replace('Follower', '')
df['Followers'] = df['Followers'].str.replace('-', '0')

# Changing str values(counts) to integers
df[['Reviews', 'Followers']] = df[['Reviews', 'Followers']].astype(int)
df[['Reviews', 'Followers']].head(5)
```

Out[21]:

	Reviews	Followers
0	1	2
1	3	2
2	2	3
3	1	1
4	3	2

In [22]:
```
# Dropping the initial column
df.drop(['Metadata'], axis=1, inplace=True)
df.head(5)
```

Out[22]:

	Name	Review	Rating	Cost	Cuisines	Timings	Reviews	Followers
0	Beyond Flavours	The ambience was good, food was quite good . h...	5.0	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)	1	2
1	Beyond Flavours	Ambience is too good for a pleasant evening. S...	5.0	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)	3	2
2	Beyond Flavours	A must try.. great food great ambience. Thnx f...	5.0	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)	2	3
3	Beyond Flavours	Soumen das and Arun was a great guy. Only beca...	5.0	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)	1	1
4	Beyond Flavours	Food is good.we ordered Kodi drumsticks and ba...	5.0	800	Chinese, Continental, Kebab, European, South I...	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)	3	2

In [23]:
```
# Sorting restaurants with their names and costs
df = df.sort_values(['Name', 'Cost'], ascending=False).reset_index()
df.drop('index', axis=1, inplace=True)
```

In [24]:
```
df.head()
```

Out[24]:

	Name	Review	Rating	Cost	Cuisines	Timings	Reviews	Followers
0	eat.fit	I had ordered gobi methi paratha.. it was ok. ...	3.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	1	1
1	eat.fit	Food was good but it was all leaking from the ...	3.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	1	0
2	eat.fit	Intially, yes,the food was really good they we...	3.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	9	0
3	eat.fit	Hyderabad's most worst and ugliest biryani i h...	1.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	1	1
4	eat.fit	Very good at quality guys..even packaging is e...	4.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	10	1

Feature Engineering - Mean of Ratings, Reviews and Followers

We find mean of the ratings given by customers and assign them to each of the restaurants

In [25]:
```
restaurants = list(df['Name'].unique())
df['Mean Rating'] = 0
df['Mean Reviews'] = 0
df['Mean Followers'] = 0

for i in range(len(restaurants)):
    df['Mean Rating'][df['Name'] == restaurants[i]] = df['Rating'][df['Name'] == restaurants[i]].mean()
    df['Mean Reviews'][df['Name'] == restaurants[i]] = df['Reviews'][df['Name'] == restaurants[i]].mean()
    df['Mean Followers'][df['Name'] == restaurants[i]] = df['Followers'][df['Name'] == restaurants[i]].mean()
```

In [26]:
```
df
```

Out[26]:

	Name	Review	Rating	Cost	Cuisines	Timings	Reviews	Followers	Mean Rating	Mean Reviews	Mean Followers
0	eat.fit	I had ordered gobi methi paratha.. it was ok. ...	3.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	1	1	3.2	27.01	191.50
1	eat.fit	Food was good but it was all leaking from the ...	3.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	1	0	3.2	27.01	191.50
2	eat.fit	Intially, yes,the food was really good they we...	3.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	9	0	3.2	27.01	191.50
3	eat.fit	Hyderabad's most worst and ugliest biryani i h...	1.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	1	1	3.2	27.01	191.50
4	eat.fit	Very good at quality guys..even packaging is e...	4.0	500	Healthy Food, North Indian, Continental, South...	7 AM to 10 PM	10	1	3.2	27.01	191.50
...	...	...	...	...	...	...	...	...	...	...	...
9995	10 Downing Street	Well I have lost count how many times I have v...	4.0	1900	North Indian, Chinese, Continental	12 Noon to 12 Midnight	8	9	3.8	39.90	245.73
9996	10 Downing Street	I have been to quite a few similar places in H...	5.0	1900	North Indian, Chinese, Continental	12 Noon to 12 Midnight	5	1	3.8	39.90	245.73
9997	10 Downing Street	Great food!! Amazing interior with a decent da...	4.0	1900	North Indian, Chinese, Continental	12 Noon to 12 Midnight	14	13	3.8	39.90	245.73
9998	10 Downing Street	In love with Calcutta 10 downing Street and fo...	5.0	1900	North Indian, Chinese, Continental	12 Noon to 12 Midnight	126	216	3.8	39.90	245.73
9999	10 Downing Street	A nice noisy eventful Saturday night happens h...	4.0	1900	North Indian, Chinese, Continental	12 Noon to 12 Midnight	22	68	3.8	39.90	245.73

10000 rows × 11 columns

Using MinMax Scaler to scale the mean rating , reviews and followers on a scale of 1-5 to avoid biasing

In [27]:
```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature_range = (1,5))
df[['Mean Rating', 'Mean Reviews', 'Mean Followers']] = scaler.fit_transform(df[['Mean Rating', 'Mean Reviews', 'Mean Followers']]).round(2)
df.sample(3)
```

Out[27]:

	Name	Review	Rating	Cost	Cuisines	Timings	Reviews	Followers	Mean Rating	Mean Reviews	Mean Followers
6817	Dunkin' Donuts	Just 5 mins from my house and been there hardl...	3.0	550	Desserts, Cafe, Beverages, Burger, Fast Food	10 AM to 11 PM	74	1134	2.12	4.16	3.94
4054	Mohammedia Shawarma	there was a lil too much oil	4.0	150	Street Food, Arabian	1 PM to 1 AM	5	1	1.95	1.10	1.02
1238	The Indi Grill	The food is excellent . The service is also ve...	5.0	1500	BBQ, Asian, Modern Indian	12noon to 11pm (Mon-Sun)	5	2	4.55	1.71	1.35

Text Preprocessing and Cleaning

We need to prepare and clean the text in 'Review' and 'Cuisines' feature in order to provide recommendation

In [28]:
```
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
```

In [29]:
```
# before text processing
df[['Review', 'Cuisines']].sample(5)
```

Out[29]:

	Review	Cuisines
1439	Loved it.\n\nThe shwarmas, the fries, the garl...	American, Wraps, Desserts
4859	Went for dinner with friends and family severa...	North Indian, Mughlai
3266	Friendly staff,ambience is so good,hygiene ,go...	Biryani, North Indian, Chinese
2658	Just because a bunch of corporate guys request...	North Indian, Chinese, Continental
8930	food is awesome, but delivered food after 1:30...	Asian

In [30]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[30]:
```
True
```

Stopwords are the English words which do not add much meaning to a sentence and can safely be ignored without sacrificing the meaning of the sentence

In [31]:
```
replace_by_space = re.compile('[/(){}\[\]\|@,;]')
remove_symbols = re.compile('[^0-9a-z #+_]')

stopwords = set(stopwords.words('english'))

def text_preprocessing(text):
    text = text.lower()
    text = replace_by_space.sub(' ', text)
    text = remove_symbols.sub('', text)

    # Remove stopwords
    text = ' '.join(word for word in text.split() if word not in stopwords)

    return text
```

In [32]:
```
df['Review'] = df['Review'].apply(text_preprocessing)
df['Cuisines'] = df['Cuisines'].apply(text_preprocessing)
```

In [33]:
```
# After processing
df[['Review','Cuisines']].sample(5)
```

Out[33]:

	Review	Cuisines
9399		american fast food salad burger
536	ulavacharu one best south indian cuisine tried...	andhra north indian chinese
5840	bad dry biryani soggy chilly chickenbetter clo...	chinese north indian
4507	name appealing saw like yes lets gobecause us ...	finger food north indian kebab chinese
9103	food ambience service everything greatgobi man...	continental

Exploratory Data Analysis (EDA)

In [34]:
```
restaurant_names = list(df['Name'].unique())
restaurant_names
```

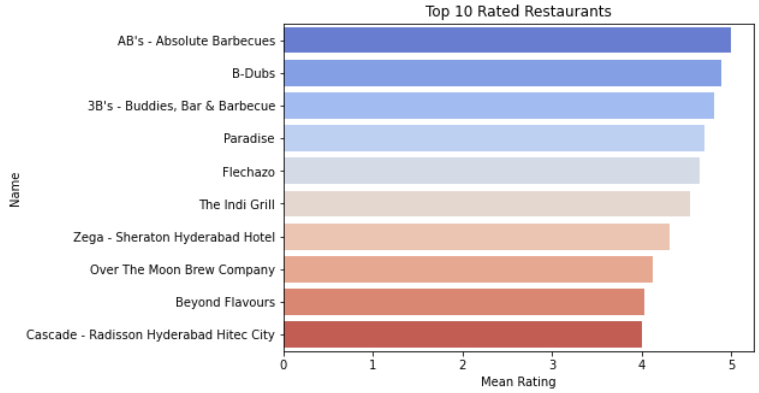
Out[34]:
```
['eat.fit',
 "Zing's Northeast Kitchen",
 'Zega - Sheraton Hyderabad Hotel',
 'Yum Yum Tree - The Arabian Food Court',
 'Urban Asia - Kitchen & Bar',
 'Ulavacharu',
 "Udipi's Upahar",
 'Triptify',
 'Tiki Shack',
 'The Tilt Bar Republic',
 'The Old Madras Baking Company',
 'The Lal Street - Bar Exchange',
 'The Indi Grill',
 'The Glass Onion',
 'The Foodie Monster Kitchen',
 "The Fisherman's Wharf",
 'The Chocolate Room',
 'Tempteys',
 'Tandoori Food Works',
 'T Grill',
 'Squeeze @ The Lime',
 'Shree Santosh Dhaba Family Restaurant',
 'Shanghai Chef 2',
 'Shah Ghouse Spl Shawarma',
 'Shah Ghouse Hotel & Restaurant',
 "Sardarji's Chaats & More",
 'SKYHY',
 'Royal Spicy Restaurant',
 'Prism Club & Kitchen',
 'PourHouse7',
 'Pot Pourri',
 'Pista House',
 'Paradise',
 'Pakwaan Grand',
 'Owm Nom Nom',
 'Over The Moon Brew Company',
 'Olive Garden',
 'NorFest - The Dhaba',
 'Mustang Terrace Lounge',
 'Momos Delight',
 'Mohammedia Shawarma',
 'Mazzo - Marriott Executive Apartments',
 'Mathura Vilas',
 'Marsala Food Company',
 'Labonel',
 'La La Land - Bar & Kitchen',
 'Kritunga Restaurant',
 'Komatose - Holiday Inn Express & Suites',
 'Khaan Saab',
 'Karachi Cafe',
 'Karachi Bakery',
 'KS Bakers',
 'KFC',
 "Jonathan's Kitchen - Holiday Inn Express & Suites",
 'Hyper Local',
 'Hyderabadi Daawat',
 'Hyderabad Chefs',
 'Hunger Maggi Point',
 'Hotel Zara Hi-Fi',
 'Hitech Bawarchi Food Zone',
 'Green Bawarchi Restaurant',
 'Gal Punjab Di',
 "GD's",
 'Frio Bistro',
 'Flechazo',
 'Feast - Sheraton Hyderabad Hotel',
 'Faasos',
 'Eat India Company',
 "Dunkin' Donuts",
 'Driven Cafe',
 "Domino's Pizza",
 'Diners Pavilion',
 'Dine O China',
 'Desi Bytes',
 'Deli 9 Bistro',
 'Delhi-39',
 'Cream Stone',
 'Collage - Hyatt Hyderabad Gachibowli',
 'Club Rogue',
 'Chinese Pavilion',
 'Cascade - Radisson Hyderabad Hitec City',
 'Cafe Eclat',
 'Biryanis And More',
 'Beyond Flavours',
 'Being Hungry',
 'Behrouz Biryani',
 'Barbeque Nation',
 'Banana Leaf Multicuisine Restaurant',
 'B-Dubs',
 'Asian Meal Box',
 'Aromas@11SIX',
 'Arena Eleven',
 'Amul',
 'American Wild Wings',
 'Al Saba Restaurant',
 'Absolute Sizzlers',
 "AB's - Absolute Barbecues",
 "3B's - Buddies, Bar & Barbecue",
```

```
 '13 Dhaba',
 '10 Downing Street']
```

Identifying top Reviewed, Rated and Followed Restaurants

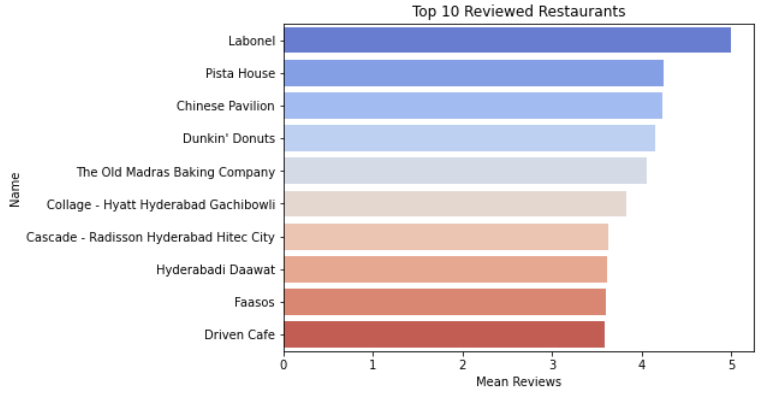
In [35]:
```
# Top 10 Rated Restaurants
df_rating = df.drop_duplicates(subset='Name')
df_rating = df_rating.sort_values(by='Mean Rating', ascending=False).head(10)
```

In [36]:
```
plt.figure(figsize=(7,5))
sns.barplot(data=df_rating, x='Mean Rating', y='Name', palette='coolwarm')
plt.title('Top 10 Rated Restaurants');
```



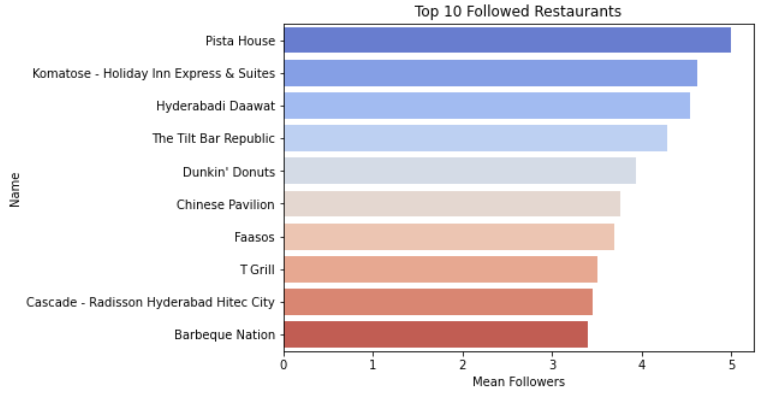
In [37]:
```
# Top 10 Reviewed Restaurants
df_reviews = df.drop_duplicates(subset='Name')
df_reviews = df_reviews.sort_values(by='Mean Reviews', ascending=False).head(10)
```

In [38]:
```
plt.figure(figsize=(7,5))
sns.barplot(data=df_reviews, x='Mean Reviews', y='Name', palette='coolwarm')
plt.title('Top 10 Reviewed Restaurants');
```



In [39]:
```
# Top 10 Followed Restaurants
df_followers = df.drop_duplicates(subset='Name')
df_followers = df_followers.sort_values(by='Mean Followers', ascending=False).head(10)
```

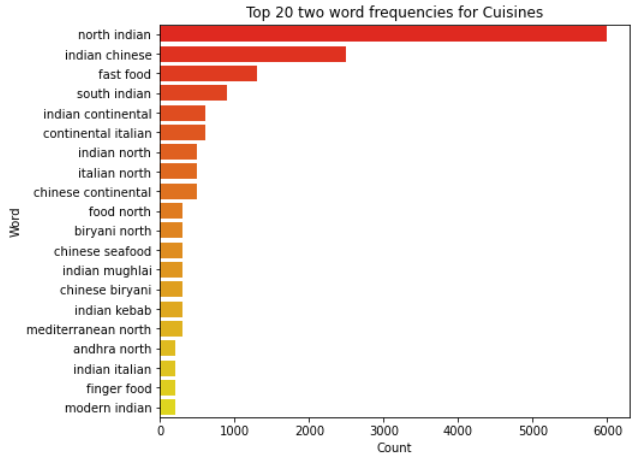
In [40]:
```
plt.figure(figsize=(7,5))
sns.barplot(data=df_followers, x='Mean Followers', y='Name', palette='coolwarm')
plt.title('Top 10 Followed Restaurants');
```



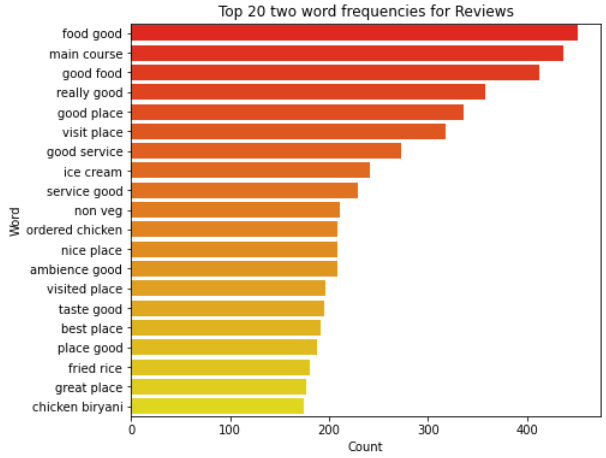
EDA - Word Frequency Distribution:

In [41]:
```
def get_top_words(column, top_n, no_of_words):

    vector = CountVectorizer(ngram_range= no_of_words, stop_words='english')

    words = vector.fit_transform(column)

    sum_words = words.sum(axis=0)

    words_freq = [(word, sum_words[0, idx]) for word, idx in vector.vocabulary_.items()]

    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)

    return words_freq[:top_n]
```

In [42]:
```
# Top 20 two-word frequencies for Cuisines
list_cuisines = get_top_words(df['Cuisines'], 20, (2,2))

df_words_cuisines = pd.DataFrame(list_cuisines, columns=['Word', 'Count'])

plt.figure(figsize=(7,6))
sns.barplot(data=df_words_cuisines, x='Count', y='Word', palette='autumn')
plt.title('Top 20 two word frequencies for Cuisines');
```



In [43]:
```
# Top 20 two-word frequencies for Reviews
list_reviews = get_top_words(df['Review'], 20, (2,2))

df_words_reviews = pd.DataFrame(list_reviews, columns=['Word', 'Count'])

plt.figure(figsize=(7,6))
sns.barplot(data=df_words_reviews, x='Count', y='Word', palette='autumn')
plt.title('Top 20 two word frequencies for Reviews');
```



TF-IDF Matrix (Term Frequency — Inverse Document Frequency Matrix)

TF-IDF -> a method used to quantify words and compute their weights. In other words, representing each word or couples of words with a number in order to use mathematics in our recommender system. Cosine similarity is a metric used to determine how similar the documents are irrespective of their size.

In [44]:
```
# Creating an index on the Name feature
df.set_index('Name', inplace=True)
```

In [45]:
```
test_df=df.groupby(['Name'])['Review'].transform(','.join)
test_df.drop_duplicates(inplace=True)
test_df.sort_values
```

Out[45]:
```
<bound method Series.sort_values of Name
eat.fit                                 ordered gobi methi paratha ok good oily much t...
Zing's Northeast Kitchen                food tooooooooo good interior ambiance cozy s...
Zega - Sheraton Hyderabad Hotel         husband visited zega dimsum festival disappoin...
Yum Yum Tree - The Arabian Food Court   6th floor act boutique building entrance gate ...
Urban Asia - Kitchen & Bar              place highly recommended working eat india com...
                                                                                  ...
Absolute Sizzlers                       service pathetic ordered sizzler lamb told lam...
AB's - Absolute Barbecues               excellent experience spiced thank krishna mona...
3B's - Buddies, Bar & Barbecue          go team dinnerthe name guys govind served us w...
13 Dhaba                                didnt go eat dhabai ordered taste amazing te i...
10 Downing Street                       ive place two times really liked ambience inte...
Name: Review, Length: 100, dtype: object>
```

In [46]:
```
indices=pd.Series(test_df.index)
```

In [47]:
```
# Creating tf-idf matrix
tfidf = TfidfVectorizer(analyzer='word', ngram_range=(1, 2), min_df=0, stop_words='english')
tfidf_matrix = tfidf.fit_transform(test_df)

# Calculating cosine similarities
cosine_similarities = linear_kernel(tfidf_matrix, tfidf_matrix)
```

In [48]:
```
print(cosine_similarities)
```

```
[[1.         0.16983605 0.15763038 ... 0.11202385 0.20160599 0.1657964 ]
 [0.16983605 1.         0.23105243 ... 0.1403301  0.14738564 0.21609719]
 [0.15763038 0.23105243 1.         ... 0.19235575 0.15156523 0.29905673]
 ...
 [0.11202385 0.1403301  0.19235575 ... 1.         0.11003223 0.18241836]
 [0.20160599 0.14738564 0.15156523 ... 0.11003223 1.         0.16786332]
 [0.1657964  0.21609719 0.29905673 ... 0.18241836 0.16786332 1.        ]]
```

In [49]:
```
cosine_similarities.shape
```

Out[49]:
```
(100, 100)
```

Creating the Recommender System

In [50]:
```
def recommend(name, cosine_similarities = cosine_similarities):

    # Create a list to put top 10 restaurants
    recommendations = []

    # Find the index of the hotel entered
    # idx = indices[indices == name].index[0]
    idx = indices[indices == name].index[0]
    # print(idx)
    # print(indices[indices == 'Paradise'])

    # Find the restaurants with a similar cosine-similarity value and sort in descending order
    score_series = pd.Series(cosine_similarities[idx]).sort_values(ascending=False)
    # print(score_series)

    # Extract top 30 restaurant indexes with a similar cosine-sim value
    top30_indexes = list(score_series.iloc[0:31].index)
    # print(top30_indexes)

    # Names of the top 30 restaurants
    for each in top30_indexes:
        recommendations.append(list(test_df.index)[each])
    # print(recommendations)

    # Creating the new data set to show similar restaurants
    df_new = pd.DataFrame(columns=['Cuisines', 'Mean Rating', 'Cost', 'Timings'])

    # Creating top 30 similar restaurants with their features
    for each in recommendations:
        df_new = df_new.append(pd.DataFrame(df[['Cuisines','Mean Rating', 'Cost', 'Timings']][df.index == each].sample()))
    # print(df_new)

    # Drop the duplicates i.e, with same restaurant names
    # Sort them based on Mean Rating in descending order and select Top 10
    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 are: ' % (str(len(df_new)), name))

    return df_new
print("Created Recommendation system")
```

```
Created Recommendation system
```

Testing the Recommender System with an example

In [51]:
```
# Details of a random restaurant
df[df.index == 'eat.fit'].head(1)
```

Out[51]:

Name	Review	Rating	Cost	Cuisines	Timings	Reviews	Followers	Mean Rating	Mean Reviews	Mean Followers
eat.fit	ordered gobi methi paratha ok good oily much t...	3.0	500	healthy food north indian continental south in...	7 AM to 10 PM	1	1	2.29	2.54	2.47

In [52]:
```
# Recommendation
recommend('Paradise')
```

```
Top 10 Restaurants like Paradise with similar reviews are:
```

Out[52]:

	Cuisines	Mean Rating	Cost	Timings
AB's - Absolute Barbecues	european mediterranean north indian	5.00	1500	12 Noon to 4:30 PM, 6:30 PM to 11:30 PM
Paradise	biryani north indian chinese	4.71	800	11 AM to 11 PM
Flechazo	asian mediterranean north indian desserts	4.65	1300	11:30 AM to 4:30 PM, 6:30 PM to 11 PM
The Indi Grill	bbq asian modern indian	4.55	1500	12noon to 11pm (Mon-Sun)
Beyond Flavours	chinese continental kebab european south india...	4.03	800	12noon to 3:30pm, 6:30pm to 11:30pm (Mon-Sun)
Barbeque Nation	mediterranean north indian kebab bbq	3.77	1600	12 Noon to 3:30 PM, 6:30 PM to 11:30 PM
PourHouse7	north indian continental chinese italian	3.35	1200	12 Noon to 12 Midnight (Mon-Thu, Sun), 12 Noon...
Deli 9 Bistro	cafe continental desserts	3.29	700	12 Noon to 10:30 PM
Hyderabad Chefs	north indian chinese	3.27	600	12 Noon to 10:30 PM
Khaan Saab	north indian mughlai	3.26	1100	12 Noon to 3:30 PM, 7 PM to 11:30 PM