
zomato

The goal of this exploratory data analysis (EDA) is to understand customer preferences and restaurant performance in Bangalore using the Zomato dataset. Specifically, this analysis aims to:

Explore the Relationship Between Ratings and Orders:

> By examining the correlation between restaurant ratings and the number of orders, we aim to uncover how customer satisfaction (via ratings) impacts the popularity and demand for restaurants in Bangalore.

Explore the preferences of people of Bangalore:

> The type of food that is most preferred by people in different regions of the city.

Perform Geospatial Analysis:

> This analysis will focus on identifying the geographical distribution of different cuisines across Bangalore. We will investigate which areas have the highest concentration of specific cuisines.

The findings will help provide insights into:

> How customer ratings influence restaurant success.
> Popular food trends in Bangalore.
> The areas in Bangalore where specific types of cuisine are most in demand, potentially helping restaurants make data-driven decisions about location and menu offerings.

Importing necessary libraries

In [103]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sqlite3
import folium
from nltk.corpus import RegexpTokenizer
from folium.plugins import HeatMap
from nltk.corpus import stopwords
from nltk import FreqDist, bigrams , trigrams
from geopy.geocoders import Nominatim
```

Reading data from a sqlite file

In [104]:
```
db =sqlite3.connect(r"C:\Users\Siddhant Ghosh\Downloads\Resources\zomato_rawdata.sqlite")
```

In [105]:
```
pd.read_sql_query("SELECT * FROM Users", db).head()
```

Out[105]:

	index	url	address	name	online_order	bo
0	0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	
1	1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	
2	2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	
3	3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	
4	4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	

Creating a Pandas dateframe using a sql quaery from the database

In [106]:
```
df = pd.read_sql_query("SELECT * FROM Users" , db)
```

In [107]:
```
df.shape
```

Out[107]:
```
(51717, 18)
```

In [108]:
```
df.columns
```

Out[108]:
```
Index(['index', 'url', 'address', 'name', 'online_order', 'book_table',
'rate',
       'votes', 'phone', 'location', 'rest_type', 'dish_liked', 'cuisine
s',
       'approx_cost(for two people)', 'reviews_list', 'menu_item',
       'listed_in(type)', 'listed_in(city)'],
      dtype='object')
```

Checking for missing values

In [109]:
```
df.isnull().sum()
```

Out[109]:
```
index                              0
url                                0
address                            0
name                               0
online_order                       0
book_table                         0
rate                            7775
votes                              0
phone                           1208
location                          21
rest_type                        227
dish_liked                     28078
cuisines                          45
approx_cost(for two people)      346
reviews_list                       0
menu_item                          0
listed_in(type)                    0
listed_in(city)                    0
dtype: int64
```

In [110]:
```
# Calculate the percentage of missing values in each column
null_percentage = (df.isnull().sum() / len(df)) * 100
# Format the result as percentages with 2 decimal places
null_percentage_formatted = null_percentage.map('{:.2f}%'.format)

print(null_percentage_formatted)
```

```
index                           0.00%
url                             0.00%
address                         0.00%
name                            0.00%
online_order                    0.00%
book_table                      0.00%
rate                           15.03%
votes                           0.00%
phone                           2.34%
location                        0.04%
rest_type                       0.44%
dish_liked                     54.29%
cuisines                        0.09%
approx_cost(for two people)     0.67%
reviews_list                    0.00%
menu_item                       0.00%
listed_in(type)                 0.00%
listed_in(city)                 0.00%
dtype: object
```

Since approximately 50% of the data would be lost by removing the missing values in the dish_liked column, we will retain this column for now.

Next, let's examine the rate column, which has 15% missing values. Given that this is a key feature, it's important to address this carefully.

In [111]:
```
df['rate'].unique()
```

Out[111]:
```
array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
       '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
       '4.3/5', 'NEW', '2.9/5', '3.5/5', None, '2.6/5', '3.8 /5', '3.4/
5',
       '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
       '3.4 /5', '-', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
       '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
       '3.5 /5', '2.7 /5', '2.5 /5', '3.2 /5', '2.6 /5', '4.5 /5',
       '4.3 /5', '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5',
       '4.9 /5', '3.0 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5',
       '2.1 /5', '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

I identified that this column contains 'NEW' and '-' values. These should be replaced with zero or np.nan.

I also noticed some entries like '3.8/5' instead of just '3.8'. We'll need to clean and standardize these values.

In [112]:
```
df['rate'].replace(('NEW' , '-') , np.nan , inplace=True)
```

In [113]:
```
df['rate']=df['rate'].apply(lambda x: float(x.split('/')[0]) if type(x)
== str else x)
```

In [114]:
```
df.rate.dtype
```

Out[114]:
```
dtype('float64')
```

In [115]:
```
df.rate
```

Out[115]:
```
0        4.1
1        4.1
2        3.8
3        3.7
4        3.8
        ...
51712    3.6
51713    NaN
51714    NaN
51715    4.3
51716    3.4
Name: rate, Length: 51717, dtype: float64
```

In [116]:
```
df['rate'].unique()
```

Out[116]:
```
array([4.1, 3.8, 3.7, 3.6, 4.6, 4. , 4.2, 3.9, 3.1, 3. , 3.2, 3.3, 2.8,
       4.4, 4.3, nan, 2.9, 3.5, 2.6, 3.4, 4.5, 2.5, 2.7, 4.7, 2.4, 2.2,
       2.3, 4.8, 4.9, 2.1, 2. , 1.8])
```

We aim to explore how many restaurants with ratings such as 0, 1, 1.2, 1.4, 1.6, and so on accept or do not accept online orders.

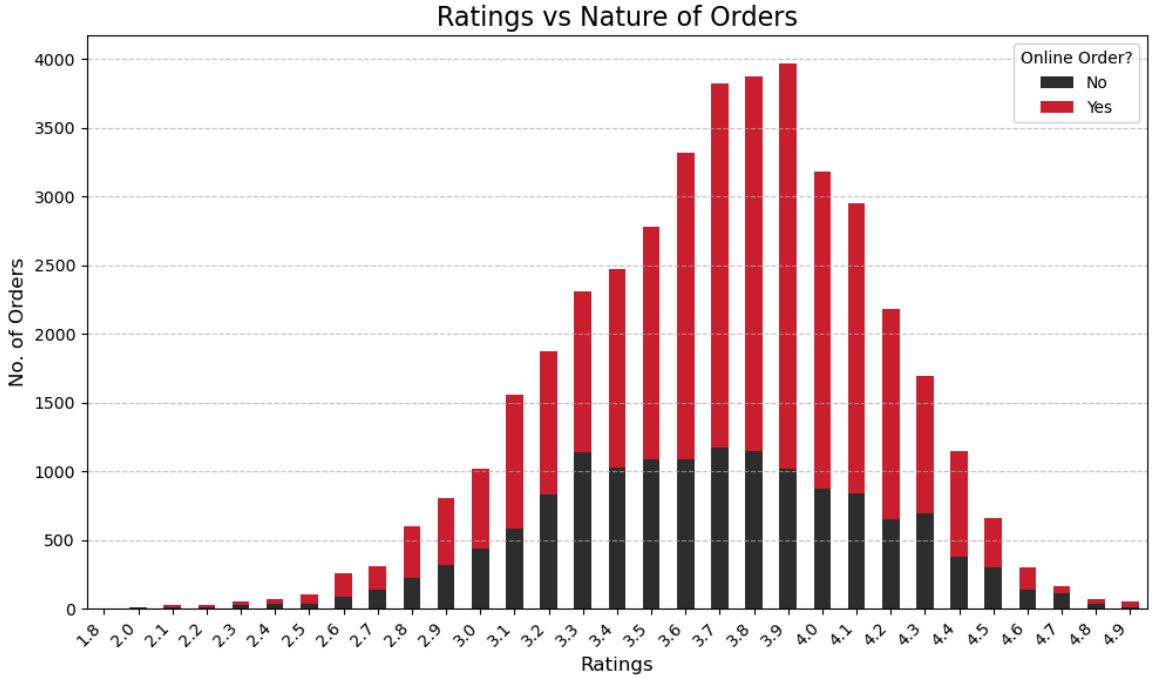
> To address this, we'll create frequency tables to capture the distribution of ratings across restaurants that accept online orders and those that don't.

In [119]:
```
x = pd.crosstab(df['rate'] , df['online_order'])
x
```

Out[119]:

online_order	No	Yes
rate		
1.8	5	0
2.0	11	0
2.1	9	15
2.2	10	16
2.3	29	22
2.4	36	34
2.5	38	63
2.6	83	177
2.7	141	166
2.8	224	376
2.9	314	488
3.0	439	584
3.1	587	974
3.2	829	1044
3.3	1137	1173
3.4	1024	1452
3.5	1090	1694
3.6	1090	2226
3.7	1172	2649
3.8	1147	2726
3.9	1017	2955
4.0	874	2309
4.1	843	2105
4.2	648	1536
4.3	692	1001
4.4	374	773
4.5	297	359
4.6	140	160
4.7	113	54
4.8	34	32
4.9	12	43

In [124]:
```
x.plot(kind='bar', stacked=True, color=['#2D2D2D','#cb202d'], figsize=(10, 6))

# Add title and labels
plt.title('Ratings vs Nature of Orders', fontsize=16)
plt.xlabel('Ratings', fontsize=12)
plt.ylabel('No. of Orders', fontsize=12)


plt.xticks(rotation=45, ha='right')

# Add a grid for better readability
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Add a legend
plt.legend(title='Online Order?', loc='upper right', fontsize=10)

# Show plot
plt.tight_layout()
plt.show()
```



We need to perform floating division of the DataFrame, or normalize the values in the x DataFrame across rows. To achieve this, we can use the x.div() function and set axis=0.

The div() function is an in-built method in pandas designed specifically for DataFrame operations.

In [125]:
```
normalize_df = x.div(x.sum(axis=1).astype(float) , axis=0)*100
```

In [126]:
```
normalize_df.plot(kind='bar', stacked=True, color=['#2D2D2D','#cb202d'],
figsize=(10, 6))

# Add title and labels
plt.title('Ratings vs Nature of Orders', fontsize=16)
plt.xlabel('Ratings', fontsize=12)
plt.ylabel('No. of Orders', fontsize=12)


plt.xticks(rotation=45, ha='right')

# Add a grid for better readability
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Add a legend
plt.legend(title='Online Order?', loc='upper right', fontsize=10)

# Show plot
plt.tight_layout()
plt.show()
```

Conclusion:

For restaurants with good ratings (i.e., greater than 4), it appears that in most cases, those that accept online orders tend to receive a higher number of ratings compared to restaurants that do not offer online ordering.

Data Cleaning to perform Text Analysis

We are going to check the different kinds of restaurants we have here but first, we need to remove missing values

In [127]:
```
rest_data = df.dropna(subset=['rest_type']).reset_index(drop=True)
```

In [128]:
```
rest_data['rest_type'].value_counts()
```

Out[128]:
```
rest_type
Quick Bites                   19132
Casual Dining                 10330
Cafe                           3732
Delivery                       2604
Dessert Parlor                 2263
                              ...
Dessert Parlor, Kiosk             2
Food Court, Beverage Shop         2
Dessert Parlor, Food Court        2
Sweet Shop, Dessert Parlor        1
Quick Bites, Kiosk                1
Name: count, Length: 93, dtype: int64
```

Let's pick 'Quick Bites' type restaurants to make some inspection:

In [129]:
```
quick_bites_df = rest_data[rest_data['rest_type'].str.contains('Quick Bites')]
```

In [130]:
```
quick_bites_df.shape
```

Out[130]:
```
(20639, 18)
```

In [131]:
```
quick_bites_df.reviews_list
```

Out[131]:
```
3        [('Rated 4.0', "RATED\n  Great food and proper...
23       [('Rated 4.0', "RATED\n  So, went here with fr...
26       [('Rated 5.0', 'RATED\n  please provide some e...
31       [('Rated 1.0', "RATED\n  Worst restaurant ever...
34       [('Rated 3.0', 'RATED\n  Not worth for the mon...
                              ...
51414    [('Rated 2.0', "RATED\n  Food is not upto the ...
51415                                                   []
51416                                                   []
51417                                                   []
51418    [('Rated 5.0', "RATED\n  I was randomly lookin...
Name: reviews_list, Length: 20639, dtype: object
```

In [132]:
```
quick_bites_df.columns
```

Out[132]:
```
Index(['index', 'url', 'address', 'name', 'online_order', 'book_table',
'rate',
       'votes', 'phone', 'location', 'rest_type', 'dish_liked', 'cuisine
s',
       'approx_cost(for two people)', 'reviews_list', 'menu_item',
       'listed_in(type)', 'listed_in(city)'],
      dtype='object')
```

Text Data Pre-processing Steps:

Convert Text to Lowercase: Transform all text data to lowercase for uniformity.
Tokenization: Break down the text into individual tokens (words).
Remove Stopwords: Eliminate common stopwords (e.g., "and", "the") from the data to focus on meaningful words.
Store Data in a List: Store the processed data in a list to compute word frequency.
Plot Word Frequencies: Perform Unigram, Bigram, and Trigram analysis to visualize word frequencies and patterns.

In [133]:
```
#Transforming all text data to lowercase
quick_bites_df['reviews_list'] = quick_bites_df['reviews_list'].apply(la
mbda x:x.lower())
```

```
C:\Users\Siddhant Ghosh\AppData\Local\Temp\ipykernel_12636\1046038503.p
y:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  quick_bites_df['reviews_list'] = quick_bites_df['reviews_list'].apply
(lambda x:x.lower())
```

In [134]:
```
##  Creating a regular expression tokenizer that have only alphabets , i
e remove all the special characters

tokenizer = RegexpTokenizer("[a-zA-Z]+")
```

In [135]:
```
tokenizer
```

Out[135]:
```
RegexpTokenizer(pattern='[a-zA-Z]+', gaps=False, discard_empty=True, fla
gs=re.UNICODE|re.MULTILINE|re.DOTALL)
```

In [136]:
```
reviews_tokens = df['reviews_list'].apply(tokenizer.tokenize)
```

In [137]:
```
reviews_tokens
```

Out[137]:
```
0        [Rated, RATED, n, A, beautiful, place, to, din...
1        [Rated, RATED, n, Had, been, here, for, dinner...
2        [Rated, RATED, n, Ambience, is, not, that, goo...
3        [Rated, RATED, n, Great, food, and, proper, Ka...
4        [Rated, RATED, n, Very, good, restaurant, in, ...
                              ...
51712    [Rated, RATED, n, Food, and, service, are, inc...
51713                                                   []
51714                                                   []
51715    [Rated, RATED, n, Nice, and, friendly, place, ...
51716    [Rated, RATED, n, Great, ambience, looking, ni...
Name: reviews_list, Length: 51717, dtype: object
```

In [138]:
```
#importing stopwords of English
stop = stopwords.words('english')
```

In [139]:
```
# Adding custom words to stopwords
stop.extend(['rated' , "n" , "nan" , "x" , "RATED" , "Rated"])
```

In [140]:
```
## remove stopwords from "reviews_tokens" Series ..
reviews_tokens_clean = reviews_tokens.apply(lambda x : [token for token
in x if token not in stop])
```

In [141]:
```
reviews_tokens_clean
```

Out[141]:
```
0        [A, beautiful, place, dine, The, interiors, ta...
1        [Had, dinner, family, Turned, good, choose, su...
2        [Ambience, good, enough, pocket, friendly, caf...
3        [Great, food, proper, Karnataka, style, full, ...
4        [Very, good, restaurant, neighbourhood, Buffet...
                              ...
51712    [Food, service, incomparably, excellent, The, ...
51713                                                   []
51714                                                   []
51715    [Nice, friendly, place, staff, awesome, Ansur,...
51716    [Great, ambience, looking, nice, good, selecti...
Name: reviews_list, Length: 51717, dtype: object
```

In [142]:
```
#converting the reviews into a 2d list
rev = reviews_tokens_clean.tolist()
```

In [143]:
```
#extracting words from each reviews to count them
total_reviews=[]

for review in rev:
    for word in review:
        total_reviews.append(word)
```

In [144]:
```
total_reviews
```

Out[144]:
```
['A',
 'beautiful',
 'place',
 'dine',
 'The',
 'interiors',
 'take',
 'back',
 'Mughal',
 'era',
 'The',
 'lightings',
 'perfect',
 'We',
 'went',
 'occasion',
 'Christmas',
 'limited',
 'items',
 'available',
 'But',
 'taste',
 'service',
 'compromised',
 'The',
 'complaint',
 'breads',
 'could',
 'better',
 'Would',
 'surely',
 'like',
 'come',
 'I',
 'dinner',
 'family',
 'weekday',
 'The',
 'restaurant',
 'completely',
 'empty',
 'Ambience',
 'good',
 'good',
 'old',
 'hindi',
 'music',
 'Seating',
 'arrangement',
 'good',
 'We',
 'ordered',
 'masala',
 'papad',
 'panner',
 'baby',
 'corn',
 'starters',
 'lemon',
 'corriander',
 'soup',
```

```
'butter',
'roti',
'olive',
'chilli',
'paratha',
'Food',
'fresh',
'good',
'service',
'good',
'Good',
'family',
'hangout',
'nCheers',
'Its',
'restaurant',
'near',
'Banashankari',
'BDA',
'Me',
'along',
'office',
'friends',
'visited',
'buffet',
'unfortunately',
'provide',
'veg',
'buffet',
'On',
'inquiring',
'said',
'place',
'mostly',
'visited',
'vegetarians',
'Anyways',
'ordered',
'ala',
'carte',
'items',
'took',
'ages',
'come',
'Food',
'ok',
'ok',
'Definitely',
'visiting',
'anymore',
'We',
'went',
'weekend',
'one',
'us',
'buffet',
'two',
'us',
'took',
'Ala',
'Carte',
```

```
'Firstly',
'ambience',
'service',
'place',
'great',
'The',
'buffet',
'lot',
'items',
'good',
'good',
'We',
'Pumpkin',
'Halwa',
'intm',
'dessert',
'amazing',
'Must',
'try',
'The',
'kulchas',
'great',
'Cheers',
'The',
'best',
'thing',
'place',
'ambiance',
'Second',
'best',
'thing',
'yummy',
'food',
'We',
'try',
'buffet',
'buffet',
'food',
'disappointed',
'us',
'nTest',
'nQuality',
'nService',
'Staff',
'professional',
'friendly',
'nOverall',
'experience',
'excellent',
'nsubirmajumder',
'wixsite',
'com',
'Great',
'food',
'pleasant',
'ambience',
'Expensive',
'Coll',
'place',
'chill',
'relax',
```

```
'nService',
'really',
'good',
'friendly',
'staff',
'nFood',
'nService',
'nAmbience',
'nOverall',
'Good',
'ambience',
'tasty',
'food',
'nCheese',
'chilli',
'paratha',
'Bhutta',
'palak',
'methi',
'curry',
'good',
'combo',
'nLemon',
'Chicken',
'starters',
'must',
'try',
'item',
'nEgg',
'fried',
'rice',
'also',
'quite',
'tasty',
'nIn',
'mocktails',
'recommend',
'Alice',
'Junoon',
'Do',
'miss',
'You',
'go',
'wrong',
'Jalsa',
'Never',
'fan',
'buffet',
'thus',
'always',
'order',
'alacarte',
'Service',
'times',
'slower',
'side',
'food',
'worth',
'wait',
'Overdelighted',
'service',
```

```
 'food',
 'provided',
 'place',
 'A',
 'royal',
 'ethnic',
 'atmosphere',
 'builds',
 'strong',
 'essence',
 'India',
 'also',
 'quality',
 'taste',
 'food',
 'truly',
 'authentic',
 'I',
 'would',
 'totally',
 'recommend',
 'visit',
 'place',
 'The',
 'place',
 'nice',
 'comfortable',
 'Food',
 'wise',
 'jalea',
 'outlets',
 'maintain',
 'good',
 'standard',
 'The',
 'soya',
 'chaap',
 'standout',
 'dish',
 'Clearly',
 'one',
 'trademark',
 'dish',
 'per',
 'must',
 'try',
 'nThe',
 'concern',
 'parking',
 'It',
 'congested',
 'limited',
 'cars',
 'The',
 'basement',
 'parking',
 'steep',
 'makes',
 'cumbersome',
 'The',
 'place',
```

```
'nice',
'comfortable',
'Food',
'wise',
'jalea',
'outlets',
'maintain',
'good',
'standard',
'The',
'soya',
'chaap',
'standout',
'dish',
'Clearly',
'one',
'trademark',
'dish',
'per',
'must',
'try',
'nThe',
'concern',
'parking',
'It',
'congested',
'limited',
'cars',
'The',
'basement',
'parking',
'steep',
'makes',
'cumbersome',
'The',
'place',
'nice',
'comfortable',
'Food',
'wise',
'jalea',
'outlets',
'maintain',
'good',
'standard',
'The',
'soya',
'chaap',
'standout',
'dish',
'Clearly',
'one',
'trademark',
'dish',
'per',
'must',
'try',
'nThe',
'concern',
'parking',
'It',
```

```
'congested',
'limited',
'cars',
'The',
'basement',
'parking',
'steep',
'makes',
'cumbersome',
'Had',
'dinner',
'family',
'Turned',
'good',
'choose',
'suitable',
'ages',
'people',
'Can',
'try',
'place',
'We',
'liked',
'starters',
'Service',
'good',
'Prices',
'affordable',
'Will',
'recommend',
'restaurant',
'early',
'dinner',
'The',
'place',
'little',
'noisy',
'The',
'ambience',
'really',
'nice',
'staff',
'courteous',
'The',
'price',
'pretty',
'high',
'quantity',
'overall',
'experience',
'fine',
'The',
'quality',
'food',
'nice',
'nothing',
'extraordinary',
'They',
'also',
'buffet',
'veg',
```

```
'I',
'felt',
'good',
'little',
'expensive',
'quantity',
'serve',
'In',
'terms',
'taste',
'decent',
'There',
'nothing',
'much',
'talk',
'ambience',
'regular',
'casual',
'dining',
'restaurant',
'take',
'family',
'dinner',
'lunch',
'If',
'improve',
'quantity',
'may',
'reduce',
'price',
'bit',
'may',
'improve',
'presentation',
'food',
'might',
'Manage',
'get',
'repeat',
'customers',
'I',
'looking',
'quite',
'place',
'spend',
'time',
'family',
'well',
'wanted',
'try',
'new',
'place',
'Since',
'I',
'Banashankari',
'I',
'thought',
'trying',
'place',
'The',
'place',
```

```
 'good',
 'rating',
 'part',
 'Zomato',
 'gold',
 'So',
 'I',
 'decided',
 'try',
 'place',
 'It',
 'delite',
 'see',
 'friendly',
 'staff',
 'food',
 'ordered',
 'tasty',
 'well',
 'nFood',
 'nAmbience',
 'nFriendly',
 'staff',
 'nPocket',
 'friendly',
 'nWill',
 'definitely',
 'visit',
 'Nice',
 'place',
 'dine',
 'good',
 'ambiance',
 'Food',
 'good',
 'serving',
 'time',
 'also',
 'good',
 'neat',
 'restrooms',
 'arranged',
 'tables',
 'thing',
 'went',
 'lunch',
 'noticed',
 'kept',
 'playing',
 'one',
 'music',
 'back',
 'back',
 'little',
 'annoying',
 'Chicken',
 'biriyani',
 'good',
 'chicken',
 'fresh',
 'tender',
```

```
 'rice',
 'well',
 'cooked',
 'overall',
 'great',
 'Mutton',
 'biriyani',
 'good',
 'tasty',
 'It',
 'plenty',
 'mutton',
 'pieces',
 'This',
 'place',
 'cool',
 'good',
 'ambience',
 'slow',
 'music',
 'delicious',
 'food',
 'find',
 'peace',
 'Staff',
 'friendly',
 'maintained',
 'place',
 'clean',
 'The',
 'price',
 'average',
 'quantity',
 'food',
 'serve',
 'nThom',
 'yum',
 'Thai',
 'soup',
 'best',
 'treat',
 'mouth',
 'roti',
 'soft',
 'vilaythi',
 'paneer',
 'perfect',
 'veggie',
 'foodies',
 'rice',
 'tried',
 'burnt',
 'garlic',
 'fried',
 'rice',
 'vegetables',
 'perfect',
 'thing',
 'end',
 'Quiet',
 'good',
```

```
'family',
'type',
'place',
'calm',
'usually',
'find',
'crowd',
'panner',
'curry',
'deserts',
'tasted',
'wer',
'really',
'good',
'found',
'little',
'expensive',
'I',
'bad',
'experience',
'nI',
'know',
'la',
'carte',
'buffet',
'worst',
'They',
'gave',
'us',
'complementary',
'drink',
'momos',
'buffet',
'The',
'momos',
'really',
'good',
'nThe',
'number',
'varieties',
'first',
'disappointing',
'The',
'service',
'slow',
'They',
'refilled',
'food',
'slowly',
'The',
'starters',
'okay',
'The',
'main',
'course',
'also',
'There',
'two',
'gravies',
'roti',
'rice',
```

```
'raitha',
'They',
'chats',
'sev',
'puri',
'pan',
'puri',
'average',
'But',
'desert',
'disappointing',
'They',
'gulab',
'Jamun',
'chocolate',
'cake',
'The',
'jamun',
'cooked',
'inside',
'There',
'cold',
'blob',
'raw',
'dough',
'inside',
'The',
'chocolate',
'cake',
'also',
'really',
'hard',
'good',
'nOverall',
'buffet',
'bad',
'experience',
'Food',
'nAmbience',
'nStaff',
'nOne',
'good',
'places',
'try',
'north',
'Indian',
'food',
'depends',
'ur',
'taste',
'buds',
'Not',
'everyone',
'like',
'items',
'Specially',
'u',
'r',
'particular',
'abt',
'sweet',
```

```
 'spicy',
 'food',
 'nThere',
 'buffet',
 'available',
 'nWe',
 'ordered',
 'paneer',
 'uttar',
 'dakshin',
 'paneer',
 'kurchan',
 'amazing',
 'The',
 'Gobi',
 'hara',
 'pyaz',
 'mix',
 'veg',
 'average',
 'A',
 'decent',
 'place',
 'family',
 'lunch',
 'dinner',
 'well',
 'arranged',
 'simple',
 'manner',
 'Food',
 'tasty',
 'crew',
 'helpful',
 'understanding',
 'Great',
 'place',
 'heavy',
 'lunch',
 'Good',
 'service',
 'nThe',
 'chicken',
 'biryani',
 'undoubtedly',
 'one',
 'best',
 'I',
 'Biriyani',
 'Lassi',
 'would',
 'suggested',
 'combo',
 'Buffet',
 'talk',
 'place',
 'try',
 'according',
 'appetite',
 'A',
 'nice',
```

```
'place',
'Its',
'one',
'restaurant',
'near',
'katriguppe',
'found',
'really',
'good',
'Good',
'variety',
'Chinese',
'thai',
'dishes',
'Service',
'good',
'good',
'place',
'hangout',
'family',
'peaceful',
'place',
'noise',
'really',
'less',
'good',
'view',
'Spice',
'elephant',
'soup',
'SPL',
'almost',
'manchow',
'flavour',
'soup',
'Just',
'medium',
'spicy',
'nLasooni',
'fish',
'tikka',
'awesome',
'nI',
'remember',
'dessert',
'name',
'I',
'attached',
'photo',
'It',
'vanilla',
'ice',
'inside',
'wafers',
'Wafer',
'hell',
'hard',
'egg',
'smell',
'chewy',
'Nightmare',
```

```
'dessert',
'nTable',
'leg',
'space',
'bad',
'I',
'uncomfortable',
'whole',
'time',
'kept',
'adjusting',
'legs',
'nNo',
'parking',
'nFor',
'taste',
'felt',
'costly',
'Zomato',
'gold',
'partner',
'price',
'It',
'insane',
'They',
'really',
'nice',
'food',
'small',
'place',
'courteous',
'staff',
'cheap',
'food',
'ambience',
'Cost',
'soups',
'Starters',
'Main',
'course',
'Cost',
'two',
'us',
'Ambience',
'good',
'enough',
'pocket',
'friendly',
'cafe',
'quantity',
'good',
'desserts',
'good',
'enough',
'nWent',
'quick',
'bite',
'friends',
'nThe',
'ambience',
'corporate',
```

```
'feel',
'I',
'would',
'say',
'unique',
'nTried',
'nachos',
'pasta',
'churros',
'lasagne',
'nNachos',
'pathetic',
'Seriously',
'order',
'nPasta',
'okayish',
'nLasagne',
'good',
'nNutella',
'churros',
'best',
'nOverall',
'okayish',
'experience',
'nPeace',
'First',
'big',
'thanks',
'staff',
'Cafe',
'Very',
'polite',
'courteous',
'nI',
'mins',
'closing',
'time',
'Without',
'discomfort',
'hesitation',
'staff',
'welcomed',
'warm',
'smile',
'said',
'still',
'open',
'though',
'preparing',
'close',
'cafe',
'day',
'nQuickly',
'ordered',
'Thai',
'green',
'curry',
'served',
'rice',
'They',
'got',
```

```
 'within',
 'mins',
 'hot',
 'freshly',
 'made',
 'nIt',
 'tasty',
 'taste',
 'coconut',
 'milk',
 'Not',
 'spicy',
 'mild',
 'spicy',
 'nI',
 'saw',
 'yummy',
 'looking',
 'dessert',
 'menu',
 'go',
 'try',
 'nA',
 'good',
 ...]
```

Unigram analysis

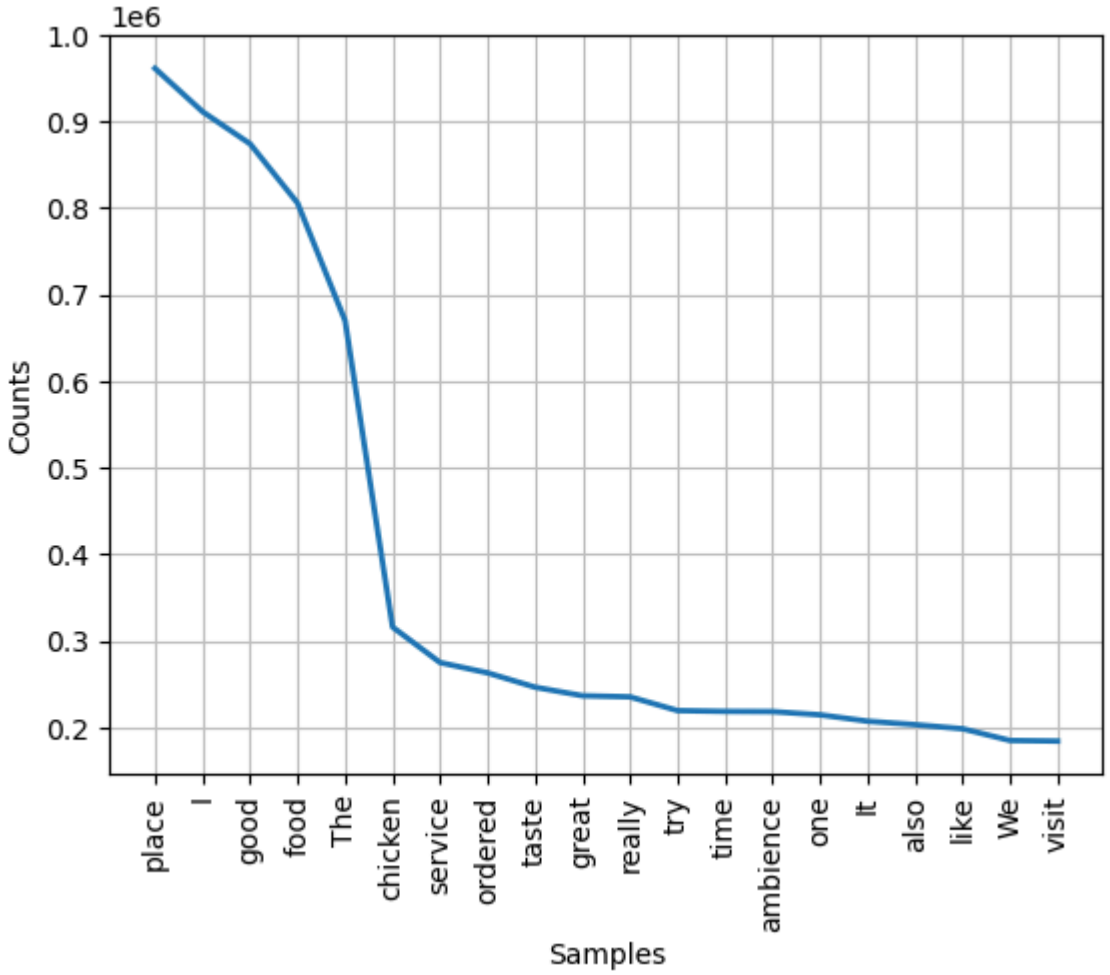
In [145]:
```
fd = FreqDist()
```

In [146]:
```
for word in total_reviews:
    fd[word] = fd[word] + 1
```

In [147]:
```
# Examining the top 20 most frequent words
fd.most_common(20)
```

Out[147]:
```
[('place', 961179),
 ('I', 911223),
 ('good', 874110),
 ('food', 805653),
 ('The', 669882),
 ('chicken', 315819),
 ('service', 274807),
 ('ordered', 262898),
 ('taste', 246338),
 ('great', 236470),
 ('really', 235171),
 ('try', 219291),
 ('time', 218394),
 ('ambience', 218136),
 ('one', 214399),
 ('It', 207054),
 ('also', 203060),
 ('like', 198316),
 ('We', 184751),
 ('visit', 184096)]
```

In [149]:
```
fd.plot(20)
```



Out[149]: <Axes: xlabel='Samples', ylabel='Counts'>

Observations

The 20 most frequent words in customer reviews include "place," "food," "good," "chicken," "taste," "service," and "ambience"
However, it's not entirely clear whether the food is actually good based solely on these words. Similarly, we need to examine the context of mentions regarding "chicken."
To derive more meaningful insights, we should consider performing a Bi-gram analysis.

Bi-gram analysis

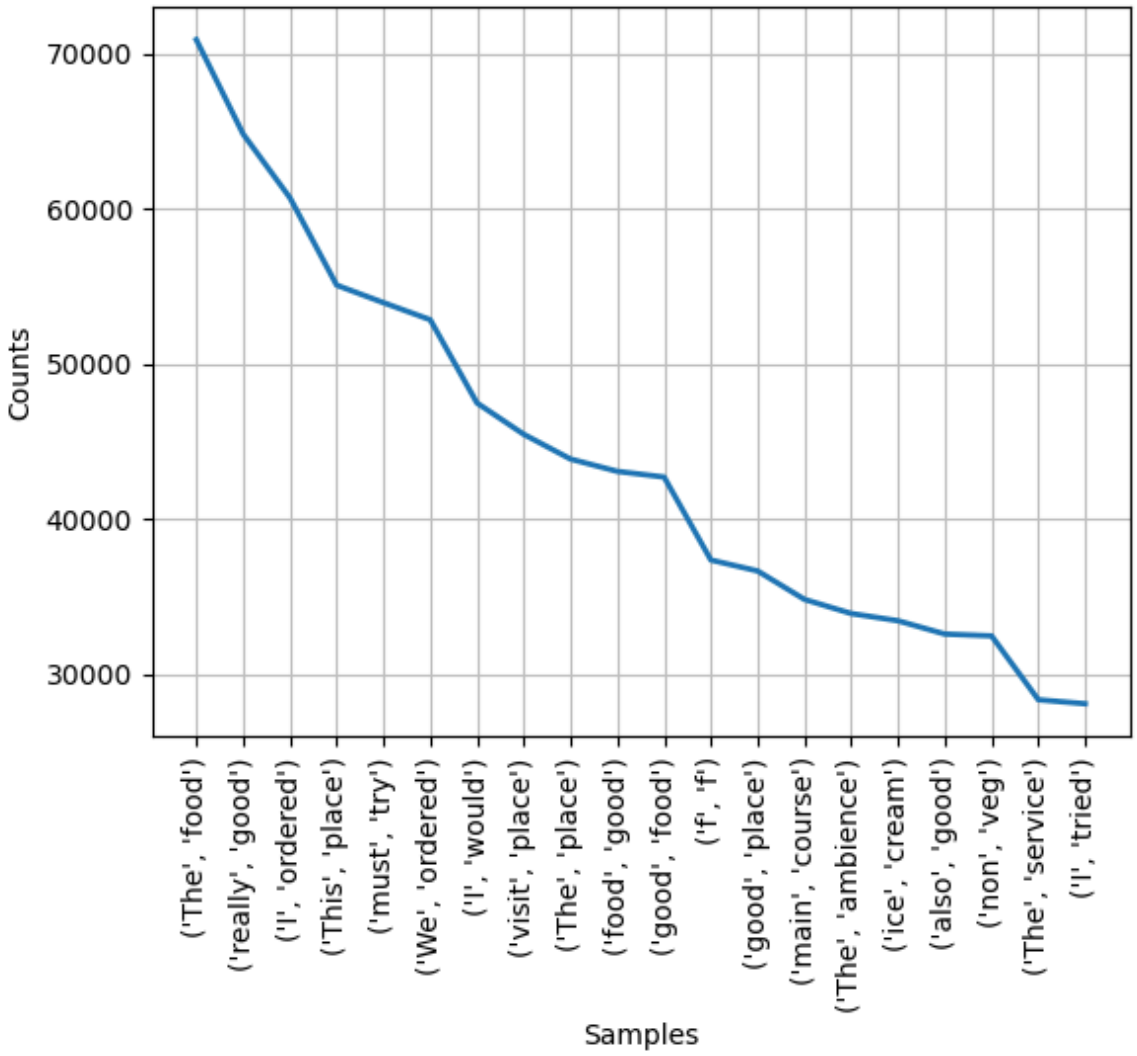
In [150]:
```
bi_grams = bigrams(total_reviews)
```

In [151]:
```
fd_bigrams = FreqDist()

for bigram in bi_grams:
    fd_bigrams[bigram] = fd_bigrams[bigram] + 1
```

In [152]:
```
fd_bigrams.most_common(20)
```

Out[152]:
```
[(('The', 'food'), 70915),
 (('really', 'good'), 64831),
 (('I', 'ordered'), 60723),
 (('This', 'place'), 55080),
 (('must', 'try'), 53951),
 (('We', 'ordered'), 52849),
 (('I', 'would'), 47479),
 (('visit', 'place'), 45456),
 (('The', 'place'), 43861),
 (('food', 'good'), 43067),
 (('good', 'food'), 42699),
 (('f', 'f'), 37348),
 (('good', 'place'), 36640),
 (('main', 'course'), 34807),
 (('The', 'ambience'), 33895),
 (('ice', 'cream'), 33429),
 (('also', 'good'), 32572),
 (('non', 'veg'), 32452),
 (('The', 'service'), 28333),
 (('I', 'tried'), 28076)]
```

In [153]:
```
fd_bigrams.plot(20)
```



Out[153]:
```
<Axes: xlabel='Samples', ylabel='Counts'>
```

In [154]:
```
fd_bigrams.most_common(100)
```

Out[154]:
```
[(('The', 'food'), 70915),
 (('really', 'good'), 64831),
 (('I', 'ordered'), 60723),
 (('This', 'place'), 55080),
 (('must', 'try'), 53951),
 (('We', 'ordered'), 52849),
 (('I', 'would'), 47479),
 (('visit', 'place'), 45456),
 (('The', 'place'), 43861),
 (('food', 'good'), 43067),
 (('good', 'food'), 42699),
 (('f', 'f'), 37348),
 (('good', 'place'), 36640),
 (('main', 'course'), 34807),
 (('The', 'ambience'), 33895),
 (('ice', 'cream'), 33429),
 (('also', 'good'), 32572),
 (('non', 'veg'), 32452),
 (('The', 'service'), 28333),
 (('I', 'tried'), 28076),
 (('place', 'I'), 27690),
 (('good', 'The'), 26858),
 (('good', 'I'), 25472),
 (('food', 'I'), 25393),
 (('one', 'best'), 24704),
 (('nWe', 'ordered'), 24624),
 (('The', 'staff'), 23892),
 (('nice', 'place'), 23531),
 (('must', 'visit'), 23505),
 (('I', 'love'), 22819),
 (('time', 'I'), 21891),
 (('place', 'good'), 21726),
 (('I', 'loved'), 21417),
 (('place', 'hangout'), 21317),
 (('pretty', 'good'), 21167),
 (('I', 'like'), 20602),
 (('quality', 'food'), 20151),
 (('service', 'good'), 20075),
 (('taste', 'good'), 20017),
 (('good', 'service'), 19724),
 (('value', 'money'), 19622),
 (('pocket', 'friendly'), 19427),
 (('I', 'visited'), 19068),
 (('fried', 'rice'), 18932),
 (('great', 'place'), 18132),
 (('nThe', 'food'), 18118),
 (('good', 'experience'), 17713),
 (('I', 'went'), 17677),
 (('visited', 'place'), 17647),
 (('ambience', 'good'), 17597),
 (('good', 'taste'), 17181),
 (('first', 'time'), 16943),
 (('recommend', 'place'), 16902),
 (('place', 'The'), 16572),
 (('I', 'really'), 16157),
 (('One', 'best'), 16069),
 (('I', 'liked'), 16047),
 (('food', 'The'), 15838),
 (('I', 'think'), 15769),
 (('really', 'nice'), 15501),
 (('food', 'great'), 15226),
```

```
(('long', 'time'), 15007),
(('We', 'went'), 14899),
(('Must', 'try'), 14793),
(('I', 'ever'), 14615),
(('North', 'Indian'), 14532),
(('chicken', 'biryani'), 14356),
(('best', 'place'), 14317),
(('nFood', 'nService'), 14285),
(('The', 'best'), 14229),
(('ordered', 'chicken'), 13943),
(('Food', 'good'), 13924),
(('nValue', 'money'), 13767),
(('good', 'ambience'), 13662),
(('Good', 'place'), 13613),
(('food', 'really'), 13395),
(('food', 'quality'), 13370),
(('I', 'got'), 12994),
(('place', 'visit'), 12988),
(('try', 'place'), 12962),
(('The', 'chicken'), 12871),
(('place', 'hang'), 12813),
(('great', 'food'), 12726),
(('quite', 'good'), 12710),
(('Nice', 'place'), 12689),
(('We', 'tried'), 12622),
(('staff', 'friendly'), 12449),
(('f', 'c'), 12404),
(('go', 'place'), 12354),
(('I', 'felt'), 12332),
(('tasted', 'good'), 12054),
(('nAmbience', 'nService'), 12017),
(('Good', 'food'), 11894),
(('food', 'service'), 11884),
(('would', 'recommend'), 11851),
(('home', 'delivery'), 11722),
(('The', 'taste'), 11710),
(('A', 'must'), 11482),
(('Indian', 'food'), 11377),
(('must', 'say'), 11275)]
```

Observations

We have gained some new insights! The food items and preferences highlighted in the top 50 bigrams include:

- Fried Rice
- North Indian
- Indian food
- Non-Veg
- Chicken Biryani
- Main Course

Factors contributing to the restaurant experience are:

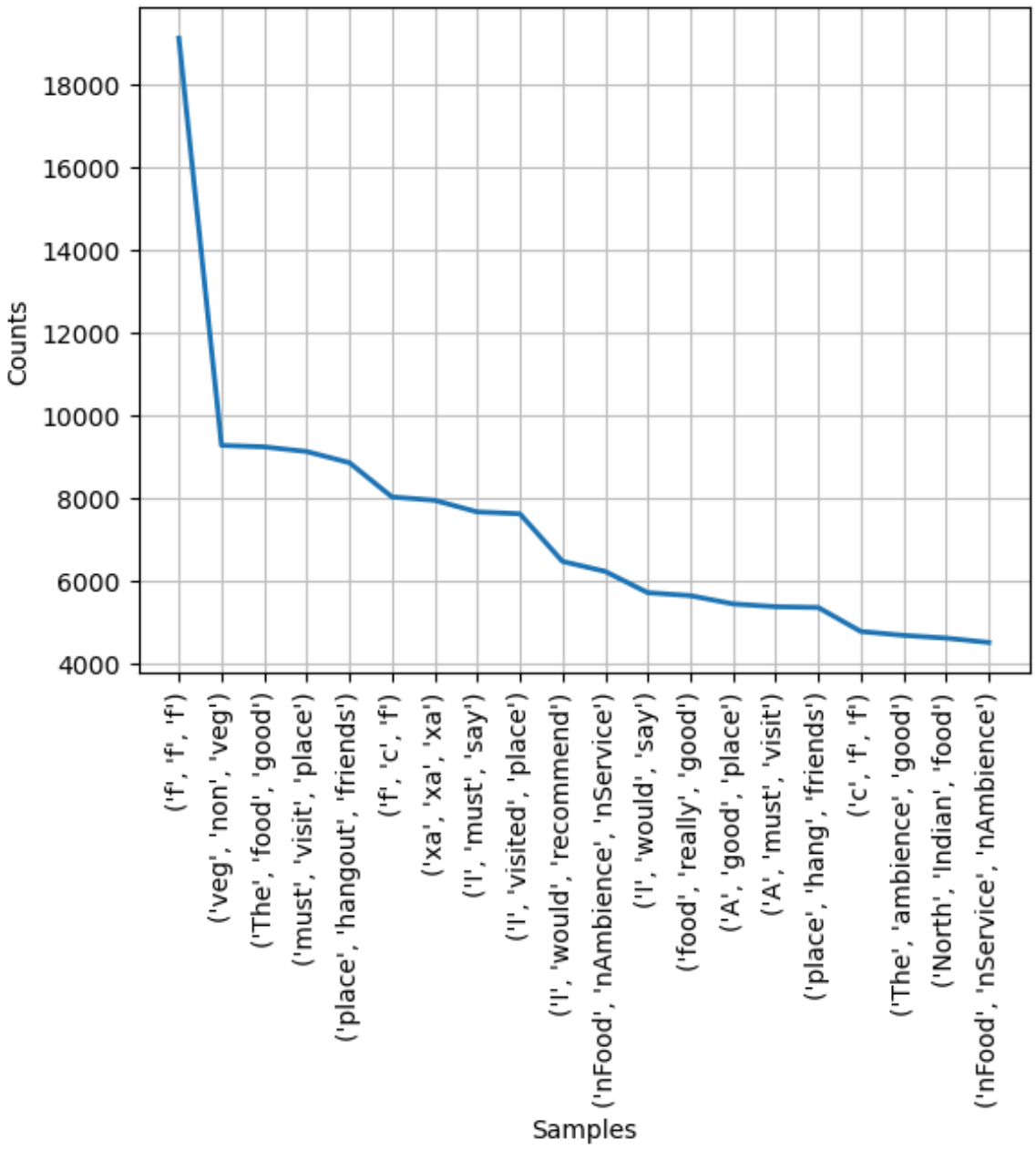
- Good Food
- Goog Service
- Pocket-Friendly
- Good Ambience
- Friendly behaviour of staffs
- Home Delivery

A key insight here is that the expense factor, which was overlooked in the individual word frequency counts, has been captured through the bigram frequency counts.

Tri-gram Analysis

In [155]:
```
tri_grams = trigrams(total_reviews)
fd_trigrams = FreqDist()

for trigram in tri_grams:
    fd_trigrams[trigram] = fd_trigrams[trigram] + 1
```

In [156]:
```
fd_trigrams.plot(20)
```



Out[156]: <Axes: xlabel='Samples', ylabel='Counts'>

In [159]:
```
fd_trigrams.most_common(100)
```

Out[159]:
```
[(('f', 'f', 'f'), 19133),
 (('veg', 'non', 'veg'), 9288),
 (('The', 'food', 'good'), 9251),
 (('must', 'visit', 'place'), 9138),
 (('place', 'hangout', 'friends'), 8869),
 (('f', 'c', 'f'), 8040),
 (('xa', 'xa', 'xa'), 7959),
 (('I', 'must', 'say'), 7678),
 (('I', 'visited', 'place'), 7633),
 (('I', 'would', 'recommend'), 6481),
 (('nFood', 'nAmbience', 'nService'), 6240),
 (('I', 'would', 'say'), 5725),
 (('food', 'really', 'good'), 5654),
 (('A', 'good', 'place'), 5454),
 (('A', 'must', 'visit'), 5386),
 (('place', 'hang', 'friends'), 5367),
 (('c', 'f', 'f'), 4787),
 (('The', 'ambience', 'good'), 4693),
 (('North', 'Indian', 'food'), 4627),
 (('nFood', 'nService', 'nAmbience'), 4521),
 (('A', 'must', 'try'), 4400),
 (('f', 'f', 'c'), 4226),
 (('The', 'food', 'really'), 4130),
 (('I', 'would', 'like'), 4129),
 (('first', 'time', 'I'), 3978),
 (('I', 'would', 'definitely'), 3947),
 (('really', 'good', 'The'), 3864),
 (('The', 'service', 'good'), 3807),
 (('nService', 'nValue', 'money'), 3782),
 (('The', 'staff', 'friendly'), 3717),
 (('nAmbience', 'nFood', 'nService'), 3708),
 (('Overall', 'good', 'experience'), 3706),
 (('The', 'staff', 'courteous'), 3543),
 (('I', 'would', 'suggest'), 3505),
 (('I', 'ordered', 'chicken'), 3474),
 (('chicken', 'fried', 'rice'), 3454),
 (('place', 'good', 'food'), 3415),
 (('must', 'try', 'place'), 3386),
 (('peri', 'peri', 'chicken'), 3371),
 (('The', 'best', 'part'), 3355),
 (('We', 'ordered', 'chicken'), 3335),
 (('nRead', 'full', 'post'), 3311),
 (('good', 'We', 'ordered'), 3248),
 (('A', 'nice', 'place'), 3205),
 (('I', 'love', 'place'), 3170),
 (('good', 'place', 'hangout'), 3148),
 (('starters', 'main', 'course'), 3141),
 (('I', 'really', 'liked'), 3131),
 (('food', 'I', 'ordered'), 3095),
 (('One', 'best', 'places'), 3023),
 (('would', 'definitely', 'recommend'), 2968),
 (('service', 'also', 'good'), 2955),
 (('definitely', 'recommend', 'place'), 2947),
 (('good', 'food', 'good'), 2886),
 (('would', 'recommend', 'place'), 2809),
 (('place', 'The', 'food'), 2793),
 (('The', 'food', 'great'), 2774),
 (('main', 'course', 'ordered'), 2756),
 (('The', 'service', 'quick'), 2731),
 (('nOverall', 'good', 'experience'), 2717),
 (('We', 'also', 'ordered'), 2715),
```

```
(('place', 'chill', 'friends'), 2698),
(('We', 'visited', 'place'), 2691),
(('nice', 'place', 'hangout'), 2690),
(('paneer', 'butter', 'masala'), 2656),
(('Keep', 'good', 'work'), 2647),
(('I', 'ordered', 'food'), 2640),
(('Must', 'visit', 'place'), 2629),
(('pocket', 'friendly', 'place'), 2614),
(('vanilla', 'ice', 'cream'), 2605),
(('A', 'great', 'place'), 2596),
(('food', 'good', 'service'), 2575),
(('place', 'I', 'would'), 2566),
(('really', 'good', 'I'), 2564),
(('place', 'good', 'ambience'), 2520),
(('food', 'We', 'ordered'), 2495),
(('north', 'Indian', 'food'), 2489),
(('place', 'really', 'good'), 2466),
(('non', 'veg', 'starters'), 2456),
(('The', 'food', 'delicious'), 2437),
(('definitely', 'visit', 'place'), 2423),
(('white', 'sauce', 'pasta'), 2419),
(('c', 'f', 'c'), 2398),
(('Overall', 'good', 'place'), 2385),
(('I', 'recommend', 'place'), 2380),
(('The', 'food', 'amazing'), 2378),
(('worth', 'every', 'penny'), 2326),
(('one', 'best', 'places'), 2310),
(('time', 'I', 'ordered'), 2296),
(('nThe', 'food', 'good'), 2284),
(('It', 'nice', 'place'), 2270),
(('The', 'ambience', 'place'), 2270),
(('f', 'b', 'f'), 2265),
(('I', 'would', 'love'), 2249),
(('I', 'would', 'rate'), 2234),
(('good', 'place', 'hang'), 2232),
(('The', 'ambience', 'great'), 2228),
(('I', 'loved', 'place'), 2220),
(('tasted', 'really', 'good'), 2219),
(('service', 'bit', 'slow'), 2215)]
```

The specific food preferences highlighted include

- North Indian food
- Paneer Butter Masala
- White Sauce Pasta
- Vanilla Ice cream
- Various chicken items.

This indicates that Bangalore is home to many chicken lovers.

In [ ]:

Extract geographical coordinates from the data

In [160]:
```
#!pip install geocoder
#!pip install geopy
```

In [161]:
```
df['location']
```

Out[161]:
```
0                       Banashankari
1                       Banashankari
2                       Banashankari
3                       Banashankari
4                       Basavanagudi
                        ...
51712                     Whitefield
51713                     Whitefield
51714                     Whitefield
51715    ITPL Main Road, Whitefield
51716    ITPL Main Road, Whitefield
Name: location, Length: 51717, dtype: object
```

In [162]:
```
df['location'].unique()
```

Out[162]:
```
array(['Banashankari', 'Basavanagudi', 'Mysore Road', 'Jayanagar',
       'Kumaraswamy Layout', 'Rajarajeshwari Nagar', 'Vijay Nagar',
       'Uttarahalli', 'JP Nagar', 'South Bangalore', 'City Market',
       'Nagarbhavi', 'Bannerghatta Road', 'BTM', 'Kanakapura Road',
       'Bommanahalli', None, 'CV Raman Nagar', 'Electronic City', 'HSR',
       'Marathahalli', 'Sarjapur Road', 'Wilson Garden', 'Shanti Nagar',
       'Koramangala 5th Block', 'Koramangala 8th Block', 'Richmond Roa
d',
       'Koramangala 7th Block', 'Jalahalli', 'Koramangala 4th Block',
       'Bellandur', 'Whitefield', 'East Bangalore', 'Old Airport Road',
       'Indiranagar', 'Koramangala 1st Block', 'Frazer Town', 'RT Naga
r',
       'MG Road', 'Brigade Road', 'Lavelle Road', 'Church Street',
       'Ulsoor', 'Residency Road', 'Shivajinagar', 'Infantry Road',
       'St. Marks Road', 'Cunningham Road', 'Race Course Road',
       'Commercial Street', 'Vasanth Nagar', 'HBR Layout', 'Domlur',
       'Ejipura', 'Jeevan Bhima Nagar', 'Old Madras Road', 'Malleshwara
m',
       'Seshadripuram', 'Kammanahalli', 'Koramangala 6th Block',
       'Majestic', 'Langford Town', 'Central Bangalore', 'Sanjay Nagar',
       'Brookefield', 'ITPL Main Road, Whitefield',
       'Varthur Main Road, Whitefield', 'KR Puram',
       'Koramangala 2nd Block', 'Koramangala 3rd Block', 'Koramangala',
       'Hosur Road', 'Rajajinagar', 'Banaswadi', 'North Bangalore',
       'Nagawara', 'Hennur', 'Kalyan Nagar', 'New BEL Road', 'Jakkur',
       'Rammurthy Nagar', 'Thippasandra', 'Kaggadasapura', 'Hebbal',
       'Kengeri', 'Sankey Road', 'Sadashiv Nagar', 'Basaveshwara Nagar',
       'Yeshwantpur', 'West Bangalore', 'Magadi Road', 'Yelahanka',
       'Sahakara Nagar', 'Peenya'], dtype=object)
```

In [163]:
```
#No. of unique locations
len(df['location'].unique())
```

Out[163]: 94

In [164]:
```
# We are adding the city, state and country to make precise analysis
df['location'] = df['location'] + " , Bangalore  , Karnataka , India "
```

In [165]:
```
df['location'].unique()
```

Out[165]:
```
array(['Banashankari , Bangalore  , Karnataka , India ',
       'Basavanagudi , Bangalore  , Karnataka , India ',
       'Mysore Road , Bangalore  , Karnataka , India ',
       'Jayanagar , Bangalore  , Karnataka , India ',
       'Kumaraswamy Layout , Bangalore  , Karnataka , India ',
       'Rajarajeshwari Nagar , Bangalore  , Karnataka , India ',
       'Vijay Nagar , Bangalore  , Karnataka , India ',
       'Uttarahalli , Bangalore  , Karnataka , India ',
       'JP Nagar , Bangalore  , Karnataka , India ',
       'South Bangalore , Bangalore  , Karnataka , India ',
       'City Market , Bangalore  , Karnataka , India ',
       'Nagarbhavi , Bangalore  , Karnataka , India ',
       'Bannerghatta Road , Bangalore  , Karnataka , India ',
       'BTM , Bangalore  , Karnataka , India ',
       'Kanakapura Road , Bangalore  , Karnataka , India ',
       'Bommanahalli , Bangalore  , Karnataka , India ', nan,
       'CV Raman Nagar , Bangalore  , Karnataka , India ',
       'Electronic City , Bangalore  , Karnataka , India ',
       'HSR , Bangalore  , Karnataka , India ',
       'Marathahalli , Bangalore  , Karnataka , India ',
       'Sarjapur Road , Bangalore  , Karnataka , India ',
       'Wilson Garden , Bangalore  , Karnataka , India ',
       'Shanti Nagar , Bangalore  , Karnataka , India ',
       'Koramangala 5th Block , Bangalore  , Karnataka , India ',
       'Koramangala 8th Block , Bangalore  , Karnataka , India ',
       'Richmond Road , Bangalore  , Karnataka , India ',
       'Koramangala 7th Block , Bangalore  , Karnataka , India ',
       'Jalahalli , Bangalore  , Karnataka , India ',
       'Koramangala 4th Block , Bangalore  , Karnataka , India ',
       'Bellandur , Bangalore  , Karnataka , India ',
       'Whitefield , Bangalore  , Karnataka , India ',
       'East Bangalore , Bangalore  , Karnataka , India ',
       'Old Airport Road , Bangalore  , Karnataka , India ',
       'Indiranagar , Bangalore  , Karnataka , India ',
       'Koramangala 1st Block , Bangalore  , Karnataka , India ',
       'Frazer Town , Bangalore  , Karnataka , India ',
       'RT Nagar , Bangalore  , Karnataka , India ',
       'MG Road , Bangalore  , Karnataka , India ',
       'Brigade Road , Bangalore  , Karnataka , India ',
       'Lavelle Road , Bangalore  , Karnataka , India ',
       'Church Street , Bangalore  , Karnataka , India ',
       'Ulsoor , Bangalore  , Karnataka , India ',
       'Residency Road , Bangalore  , Karnataka , India ',
       'Shivajinagar , Bangalore  , Karnataka , India ',
       'Infantry Road , Bangalore  , Karnataka , India ',
       'St. Marks Road , Bangalore  , Karnataka , India ',
       'Cunningham Road , Bangalore  , Karnataka , India ',
       'Race Course Road , Bangalore  , Karnataka , India ',
       'Commercial Street , Bangalore  , Karnataka , India ',
       'Vasanth Nagar , Bangalore  , Karnataka , India ',
       'HBR Layout , Bangalore  , Karnataka , India ',
       'Domlur , Bangalore  , Karnataka , India ',
       'Ejipura , Bangalore  , Karnataka , India ',
       'Jeevan Bhima Nagar , Bangalore  , Karnataka , India ',
       'Old Madras Road , Bangalore  , Karnataka , India ',
       'Malleshwaram , Bangalore  , Karnataka , India ',
       'Seshadripuram , Bangalore  , Karnataka , India ',
       'Kammanahalli , Bangalore  , Karnataka , India ',
       'Koramangala 6th Block , Bangalore  , Karnataka , India ',
       'Majestic , Bangalore  , Karnataka , India ',
       'Langford Town , Bangalore  , Karnataka , India ',
```

```
       'Central Bangalore , Bangalore  , Karnataka , India ',
       'Sanjay Nagar , Bangalore  , Karnataka , India ',
       'Brookefield , Bangalore  , Karnataka , India ',
       'ITPL Main Road, Whitefield , Bangalore  , Karnataka , India ',
       'Varthur Main Road, Whitefield , Bangalore  , Karnataka , India
    ',
       'KR Puram , Bangalore  , Karnataka , India ',
       'Koramangala 2nd Block , Bangalore  , Karnataka , India ',
       'Koramangala 3rd Block , Bangalore  , Karnataka , India ',
       'Koramangala , Bangalore  , Karnataka , India ',
       'Hosur Road , Bangalore  , Karnataka , India ',
       'Rajajinagar , Bangalore  , Karnataka , India ',
       'Banaswadi , Bangalore  , Karnataka , India ',
       'North Bangalore , Bangalore  , Karnataka , India ',
       'Nagawara , Bangalore  , Karnataka , India ',
       'Hennur , Bangalore  , Karnataka , India ',
       'Kalyan Nagar , Bangalore  , Karnataka , India ',
       'New BEL Road , Bangalore  , Karnataka , India ',
       'Jakkur , Bangalore  , Karnataka , India ',
       'Rammurthy Nagar , Bangalore  , Karnataka , India ',
       'Thippasandra , Bangalore  , Karnataka , India ',
       'Kaggadasapura , Bangalore  , Karnataka , India ',
       'Hebbal , Bangalore  , Karnataka , India ',
       'Kengeri , Bangalore  , Karnataka , India ',
       'Sankey Road , Bangalore  , Karnataka , India ',
       'Sadashiv Nagar , Bangalore  , Karnataka , India ',
       'Basaveshwara Nagar , Bangalore  , Karnataka , India ',
       'Yeshwantpur , Bangalore  , Karnataka , India ',
       'West Bangalore , Bangalore  , Karnataka , India ',
       'Magadi Road , Bangalore  , Karnataka , India ',
       'Yelahanka , Bangalore  , Karnataka , India ',
       'Sahakara Nagar , Bangalore  , Karnataka , India ',
       'Peenya , Bangalore  , Karnataka , India '], dtype=object)
```

In [166]:
```
df_copy = df.copy()
```

In [167]:
```
df_copy = df_copy.dropna(subset=['location'])
```

In [168]:
```
locations = pd.DataFrame(df_copy['location'].unique())
```

In [169]:
```
locations
```

Out[169]:

	0
0	Banashankari , Bangalore , Karnataka , India
1	Basavanagudi , Bangalore , Karnataka , India
2	Mysore Road , Bangalore , Karnataka , India
3	Jayanagar , Bangalore , Karnataka , India
4	Kumaraswamy Layout , Bangalore , Karnataka , ...
...	...
88	West Bangalore , Bangalore , Karnataka , India
89	Magadi Road , Bangalore , Karnataka , India
90	Yelahanka , Bangalore , Karnataka , India
91	Sahakara Nagar , Bangalore , Karnataka , India
92	Peenya , Bangalore , Karnataka , India

93 rows × 1 columns

In [170]:
```
#Naming the column
locations.columns = ['Area']
```

In [171]:
```
from geopy.geocoders import Nominatim
```

In [172]:
```
### assign timeout=None in order to get rid of timeout error..
geolocator = Nominatim(user_agent="app" , timeout=None)
```

In [173]:
```
#to assign co-ordinates into a list
lat=[]
lon=[]

for location in locations['Area']:
    location = geolocator.geocode(location)
    if location is None:
        lat.append(np.nan)
        lon.append(np.nan)
    else:
        lat.append(location.latitude)
        lon.append(location.longitude)
```

In [174]:
```
# adding the latitude and longitude column in the locations dataset
locations['latitude'] = lat
locations['longitude'] = lon
```

In [175]:
```
locations
```

Out[175]:

	Area	latitude	longitude
0	Banashankari , Bangalore , Karnataka , India	12.939333	77.553982
1	Basavanagudi , Bangalore , Karnataka , India	12.941726	77.575502
2	Mysore Road , Bangalore , Karnataka , India	12.953672	77.542512
3	Jayanagar , Bangalore , Karnataka , India	12.939904	77.582638
4	Kumaraswamy Layout , Bangalore , Karnataka , ...	12.906768	77.559502
...	...	...	...
88	West Bangalore , Bangalore , Karnataka , India	13.009652	77.553054
89	Magadi Road , Bangalore , Karnataka , India	12.975653	77.555355
90	Yelahanka , Bangalore , Karnataka , India	13.107915	77.585524
91	Sahakara Nagar , Bangalore , Karnataka , India	13.062147	77.580061
92	Peenya , Bangalore , Karnataka , India	13.032942	77.527325

93 rows × 3 columns

In [176]:
```
### lets find it out whether we have misssing values or not !
locations.isnull().sum()
```

Out[176]:
```
Area         0
latitude     2
longitude    2
dtype: int64
```

In [177]:
```
locations[locations['latitude'].isna()]
```

Out[177]:

	Area	latitude	longitude
79	Rammurthy Nagar , Bangalore , Karnataka , India	NaN	NaN
85	Sadashiv Nagar , Bangalore , Karnataka , India	NaN	NaN

In [178]:
```
#Adding missing values from google search
locations['latitude'][79] = 13.0163
locations['longitude'][79] = 77.6785
locations['latitude'][85] = 13.0068
locations['longitude'][85] = 77.5813
```

```
C:\Users\Siddhant Ghosh\AppData\Local\Temp\ipykernel_12636\1720192873.p
y:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  locations['latitude'][79] = 13.0163
C:\Users\Siddhant Ghosh\AppData\Local\Temp\ipykernel_12636\1720192873.p
y:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  locations['longitude'][79] = 77.6785
C:\Users\Siddhant Ghosh\AppData\Local\Temp\ipykernel_12636\1720192873.p
y:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  locations['latitude'][85] = 13.0068
C:\Users\Siddhant Ghosh\AppData\Local\Temp\ipykernel_12636\1720192873.p
y:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  locations['longitude'][85] = 77.5813
```

In [179]:
```
df['cuisines'].isnull().sum()
```

Out[179]:
```
45
```

In [180]:
```
df = df.dropna(subset=['cuisines'])
```

In [181]:
```
df.cuisines.unique()
```

Out[181]:
```
array(['North Indian, Mughlai, Chinese', 'Chinese, North Indian, Thai',
       'Cafe, Mexican, Italian', ...,
       'North Indian, Street Food, Biryani', 'Chinese, Mughlai',
       'North Indian, Chinese, Arabian, Momos'], dtype=object)
```

In [182]:
```
#Let's find it out what are those areas where we have most number of Mom
os restaurants ?
#Because I love them

momos = df[df['cuisines'].str.contains('Momo')]
```

In [183]:
```
momos.shape
```

Out[183]:
```
(1276, 18)
```

In [184]:
```
momos_rest_count = momos['location'].value_counts().reset_index().rename
(columns={'index':'name' , "location":"Area"})
```

In [185]:
```
momos_rest_count
```

Out[185]:

	Area	count
0	Koramangala 5th Block , Bangalore , Karnataka...	169
1	BTM , Bangalore , Karnataka , India	105
2	Indiranagar , Bangalore , Karnataka , India	68
3	Brigade Road , Bangalore , Karnataka , India	55
4	Jayanagar , Bangalore , Karnataka , India	52
5	Bannerghatta Road , Bangalore , Karnataka , I...	49
6	HSR , Bangalore , Karnataka , India	49
7	Whitefield , Bangalore , Karnataka , India	44
8	Marathahalli , Bangalore , Karnataka , India	42
9	Banashankari , Bangalore , Karnataka , India	39
10	JP Nagar , Bangalore , Karnataka , India	34
11	Koramangala 7th Block , Bangalore , Karnataka...	34
12	Frazer Town , Bangalore , Karnataka , India	30
13	Church Street , Bangalore , Karnataka , India	28
14	Koramangala 8th Block , Bangalore , Karnataka...	27
15	Ejipura , Bangalore , Karnataka , India	25
16	Koramangala 4th Block , Bangalore , Karnataka...	24
17	Kammanahalli , Bangalore , Karnataka , India	23
18	Banaswadi , Bangalore , Karnataka , India	23
19	Cunningham Road , Bangalore , Karnataka , India	22
20	Kalyan Nagar , Bangalore , Karnataka , India	21
21	Koramangala 1st Block , Bangalore , Karnataka...	20
22	Ulsoor , Bangalore , Karnataka , India	20
23	Koramangala 6th Block , Bangalore , Karnataka...	20
24	Sarjapur Road , Bangalore , Karnataka , India	19
25	Lavelle Road , Bangalore , Karnataka , India	19
26	Residency Road , Bangalore , Karnataka , India	19
27	New BEL Road , Bangalore , Karnataka , India	16
28	Malleshwaram , Bangalore , Karnataka , India	14
29	Commercial Street , Bangalore , Karnataka , I...	13
30	Electronic City , Bangalore , Karnataka , India	13
31	Domlur , Bangalore , Karnataka , India	11
32	Brookefield , Bangalore , Karnataka , India	11
33	Bellandur , Bangalore , Karnataka , India	10
34	Basavanagudi , Bangalore , Karnataka , India	9
35	Kumaraswamy Layout , Bangalore , Karnataka , ...	9
36	Old Airport Road , Bangalore , Karnataka , In...	8

	Area	count
37	MG Road , Bangalore , Karnataka , India	8
38	Richmond Road , Bangalore , Karnataka , India	7
39	Bommanahalli , Bangalore , Karnataka , India	7
40	Seshadripuram , Bangalore , Karnataka , India	6
41	Rajajinagar , Bangalore , Karnataka , India	6
42	South Bangalore , Bangalore , Karnataka , India	6
43	Kaggadasapura , Bangalore , Karnataka , India	5
44	HBR Layout , Bangalore , Karnataka , India	5
45	Nagawara , Bangalore , Karnataka , India	4
46	Jeevan Bhima Nagar , Bangalore , Karnataka , ...	4
47	Thippasandra , Bangalore , Karnataka , India	3
48	Varthur Main Road, Whitefield , Bangalore , K...	3
49	ITPL Main Road, Whitefield , Bangalore , Karn...	3
50	Basaveshwara Nagar , Bangalore , Karnataka , ...	3
51	Hennur , Bangalore , Karnataka , India	2
52	Sadashiv Nagar , Bangalore , Karnataka , India	2
53	Sanjay Nagar , Bangalore , Karnataka , India	2
54	Yeshwantpur , Bangalore , Karnataka , India	1
55	Old Madras Road , Bangalore , Karnataka , India	1
56	CV Raman Nagar , Bangalore , Karnataka , India	1
57	Sahakara Nagar , Bangalore , Karnataka , India	1
58	KR Puram , Bangalore , Karnataka , India	1
59	RT Nagar , Bangalore , Karnataka , India	1

In [186]:
```
heatmap_df = momos_rest_count.merge(locations , on='Area' , how='left')
```

In [187]:
```
heatmap_df
```

Out[187]:

	Area	count	latitude	longitude
0	Koramangala 5th Block , Bangalore , Karnataka...	169	12.934843	77.618977
1	BTM , Bangalore , Karnataka , India	105	12.916360	77.604733
2	Indiranagar , Bangalore , Karnataka , India	68	12.996298	77.545278
3	Brigade Road , Bangalore , Karnataka , India	55	12.966322	77.606805
4	Jayanagar , Bangalore , Karnataka , India	52	12.939904	77.582638
5	Bannerghatta Road , Bangalore , Karnataka , I...	49	12.952721	77.605161
6	HSR , Bangalore , Karnataka , India	49	12.900563	77.649475
7	Whitefield , Bangalore , Karnataka , India	44	12.969637	77.749745
8	Marathahalli , Bangalore , Karnataka , India	42	12.955257	77.698416
9	Banashankari , Bangalore , Karnataka , India	39	12.939333	77.553982
10	JP Nagar , Bangalore , Karnataka , India	34	12.909694	77.586607
11	Koramangala 7th Block , Bangalore , Karnataka...	34	12.936485	77.613478
12	Frazer Town , Bangalore , Karnataka , India	30	12.998683	77.615525
13	Church Street , Bangalore , Karnataka , India	28	12.975628	77.602366
14	Koramangala 8th Block , Bangalore , Karnataka...	27	12.940869	77.617338
15	Ejipura , Bangalore , Karnataka , India	25	12.942709	77.630714
16	Koramangala 4th Block , Bangalore , Karnataka...	24	12.932778	77.629405
17	Kammanahalli , Bangalore , Karnataka , India	23	13.009346	77.637709
18	Banaswadi , Bangalore , Karnataka , India	23	13.005836	77.628132
19	Cunningham Road , Bangalore , Karnataka , India	22	12.985592	77.596157
20	Kalyan Nagar , Bangalore , Karnataka , India	21	13.022142	77.640337
21	Koramangala 1st Block , Bangalore , Karnataka...	20	12.927725	77.632782
22	Ulsoor , Bangalore , Karnataka , India	20	12.977879	77.624670
23	Koramangala 6th Block , Bangalore , Karnataka...	20	12.939025	77.623848
24	Sarjapur Road , Bangalore , Karnataka , India	19	12.924009	77.653110
25	Lavelle Road , Bangalore , Karnataka , India	19	12.974949	77.599725
26	Residency Road , Bangalore , Karnataka , India	19	12.966575	77.598623
27	New BEL Road , Bangalore , Karnataka , India	16	13.021940	77.572312
28	Malleshwaram , Bangalore , Karnataka , India	14	13.002735	77.570325
29	Commercial Street , Bangalore , Karnataka , I...	13	12.982232	77.608295
30	Electronic City , Bangalore , Karnataka , India	13	12.848760	77.648253
31	Domlur , Bangalore , Karnataka , India	11	12.962467	77.638196
32	Brookefield , Bangalore , Karnataka , India	11	12.963814	77.722437
33	Bellandur , Bangalore , Karnataka , India	10	12.931032	77.678247
34	Basavanagudi , Bangalore , Karnataka , India	9	12.941726	77.575502
35	Kumaraswamy Layout , Bangalore , Karnataka , ...	9	12.906768	77.559502
36	Old Airport Road , Bangalore , Karnataka , In...	8	12.960662	77.642263

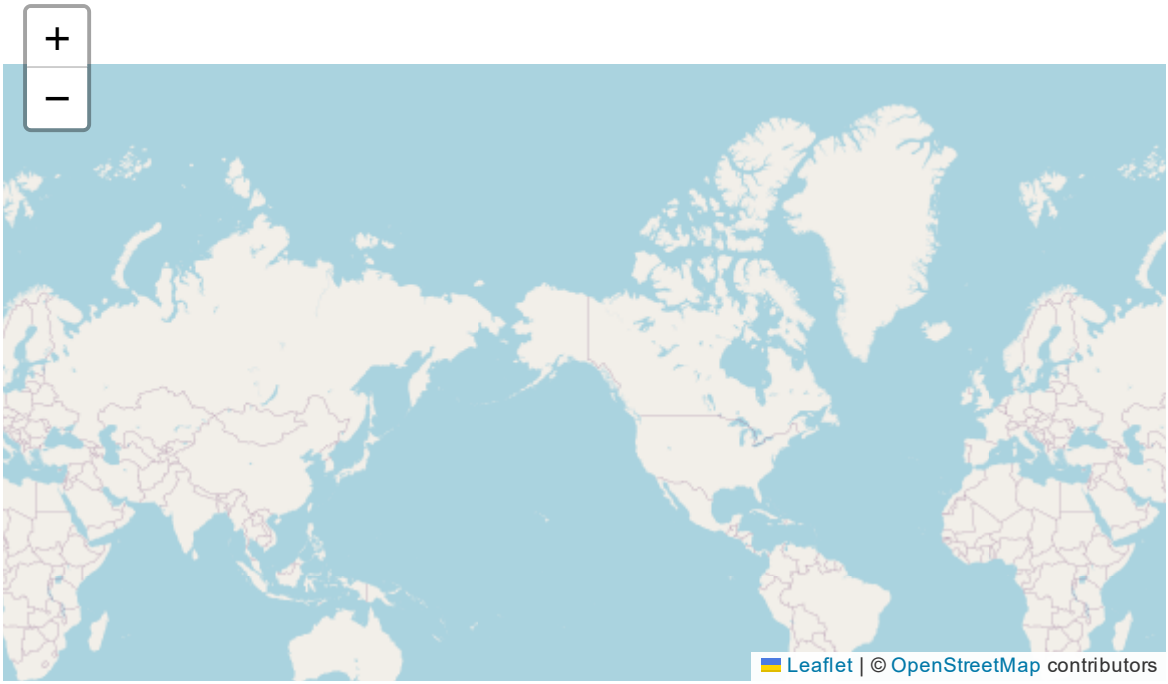
	Area	count	latitude	longitude
37	MG Road , Bangalore , Karnataka , India	8	12.975526	77.606790
38	Richmond Road , Bangalore , Karnataka , India	7	12.966237	77.606756
39	Bommanahalli , Bangalore , Karnataka , India	7	12.908945	77.623904
40	Seshadripuram , Bangalore , Karnataka , India	6	12.993188	77.575342
41	Rajajinagar , Bangalore , Karnataka , India	6	13.000536	77.549700
42	South Bangalore , Bangalore , Karnataka , India	6	12.864107	77.554416
43	Kaggadasapura , Bangalore , Karnataka , India	5	12.984671	77.679091
44	HBR Layout , Bangalore , Karnataka , India	5	13.035870	77.632360
45	Nagawara , Bangalore , Karnataka , India	4	13.042279	77.624858
46	Jeevan Bhima Nagar , Bangalore , Karnataka , ...	4	12.967807	77.656837
47	Thippasandra , Bangalore , Karnataka , India	3	12.973936	77.650998
48	Varthur Main Road, Whitefield , Bangalore , K...	3	12.962176	77.771367
49	ITPL Main Road, Whitefield , Bangalore , Karn...	3	12.987639	77.737772
50	Basaveshwara Nagar , Bangalore , Karnataka , ...	3	12.991180	77.544892
51	Hennur , Bangalore , Karnataka , India	2	13.025809	77.630507
52	Sadashiv Nagar , Bangalore , Karnataka , India	2	13.006800	77.581300
53	Sanjay Nagar , Bangalore , Karnataka , India	2	12.957866	77.695875
54	Yeshwantpur , Bangalore , Karnataka , India	1	13.023830	77.552921
55	Old Madras Road , Bangalore , Karnataka , India	1	12.985725	77.645128
56	CV Raman Nagar , Bangalore , Karnataka , India	1	12.985099	77.663117
57	Sahakara Nagar , Bangalore , Karnataka , India	1	13.062147	77.580061
58	KR Puram , Bangalore , Karnataka , India	1	13.007516	77.695935
59	RT Nagar , Bangalore , Karnataka , India	1	13.022720	77.595715

Adding Basemap

In [188]:
```
#!pip install folium
import folium
basemap = folium.Map()
```

In [189]:
```
basemap
```

Out[189]:



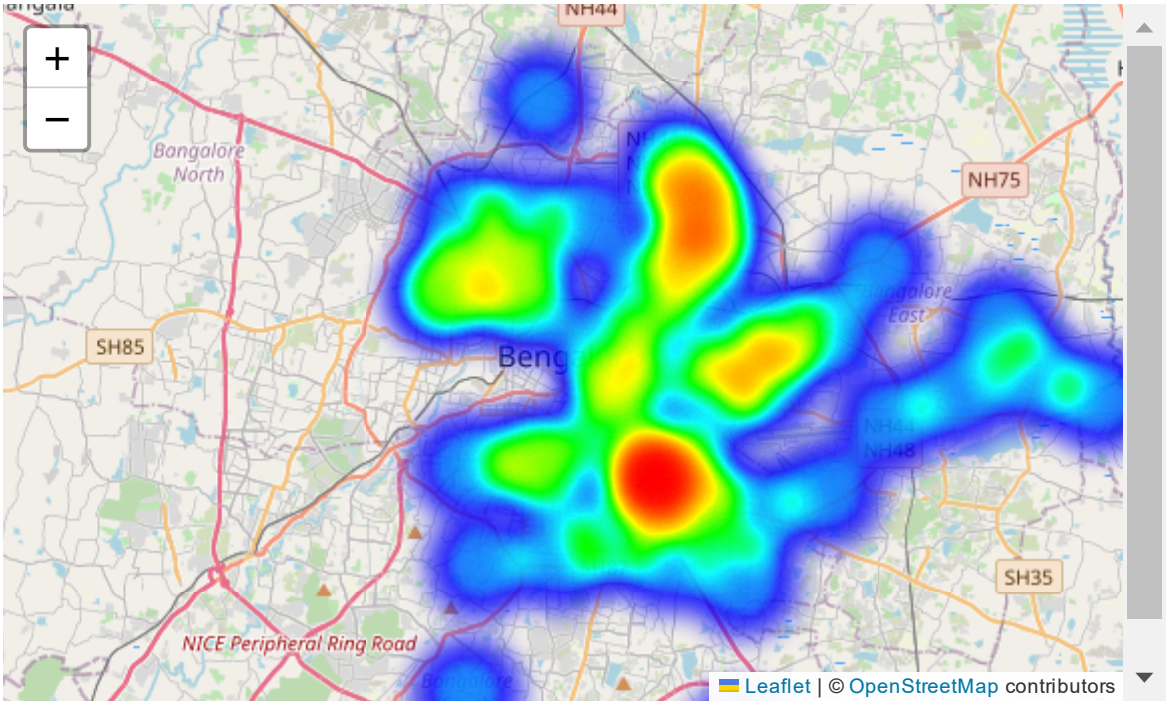
In [190]:
```
from folium.plugins import HeatMap
```

In [200]:
```
HeatMap(heatmap_df[['latitude', 'longitude' , "count"]]).add_to(basemap)
```

Out[200]:
```
<folium.plugins.heat_map.HeatMap at 0x2547ca30450>
```

In [201]:
```
basemap
```

Out[201]:



Conclusions:

- It is evident that restaurants are primarily concentrated in the central Bangalore area.
- The density of restaurants decreases as we move away from the center.
- This information can be valuable for potential restaurant entrepreneurs in identifying favorable locations for their ventures.
- It's important to note that heatmaps are most effective when we have latitude and longitude data or when indicating the significance or count of specific locations.

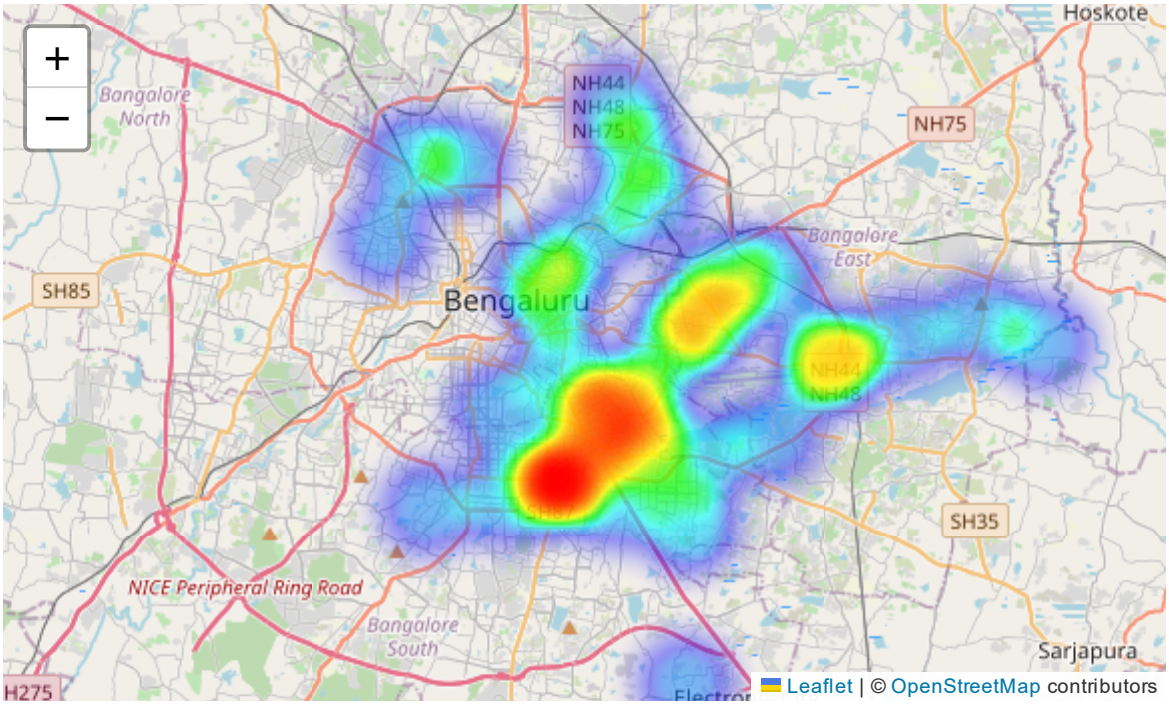
Automating the task of generating basemap for different type of cuisnes

In [193]:
```
def get_heatmap(cuisine):
    cuisine_df = df[df['cuisines'].str.contains(cuisine)]

    cuisine_rest_count = cuisine_df['location'].value_counts().reset_ind
ex().rename(columns={'index':'name' , "location":"Area"})
    heatmap_df = cuisine_rest_count.merge(locations , on='Area' , how='l
eft')
    print(heatmap_df.head(4))

    basemap = folium.Map()
    HeatMap(heatmap_df[['latitude', 'longitude' , "count"]]).add_to(base
map)
    return basemap
```

In [194]:
```
#Let's find Bengali restaurants as I'm a Bengali and I'd love to find a place which serves bengali food
get_heatmap('Bengali')
```

```
                                                  Area  count   latitude
\
0                BTM , Bangalore  , Karnataka , India      79  12.916360
1     Marathahalli , Bangalore  , Karnataka , India      52  12.955257
2  Koramangala 1st Block , Bangalore  , Karnataka...     46  12.927725
3                HSR , Bangalore  , Karnataka , India      45  12.900563

   longitude
0  77.604733
1  77.698416
2  77.632782
3  77.649475
```

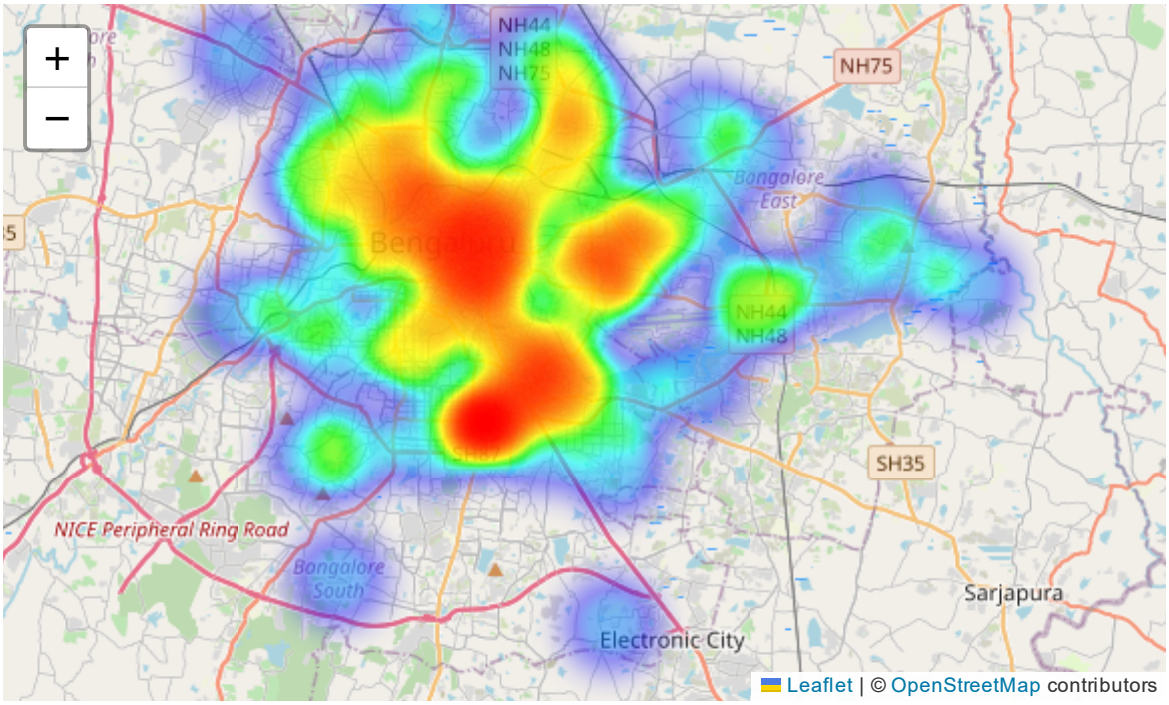
Out[194]:



In [195]:
```
#for South Indian cuisines
get_heatmap('South Indian')
```

```
                                                Area  count   latitude  longit
ude
0         BTM , Bangalore  , Karnataka , India     815  12.916360  77.604
733
1    JP Nagar , Bangalore  , Karnataka , India     437  12.909694  77.586
607
2         HSR , Bangalore  , Karnataka , India     436  12.900563  77.649
475
3   Jayanagar , Bangalore  , Karnataka , India     416  12.939904  77.582
638
```

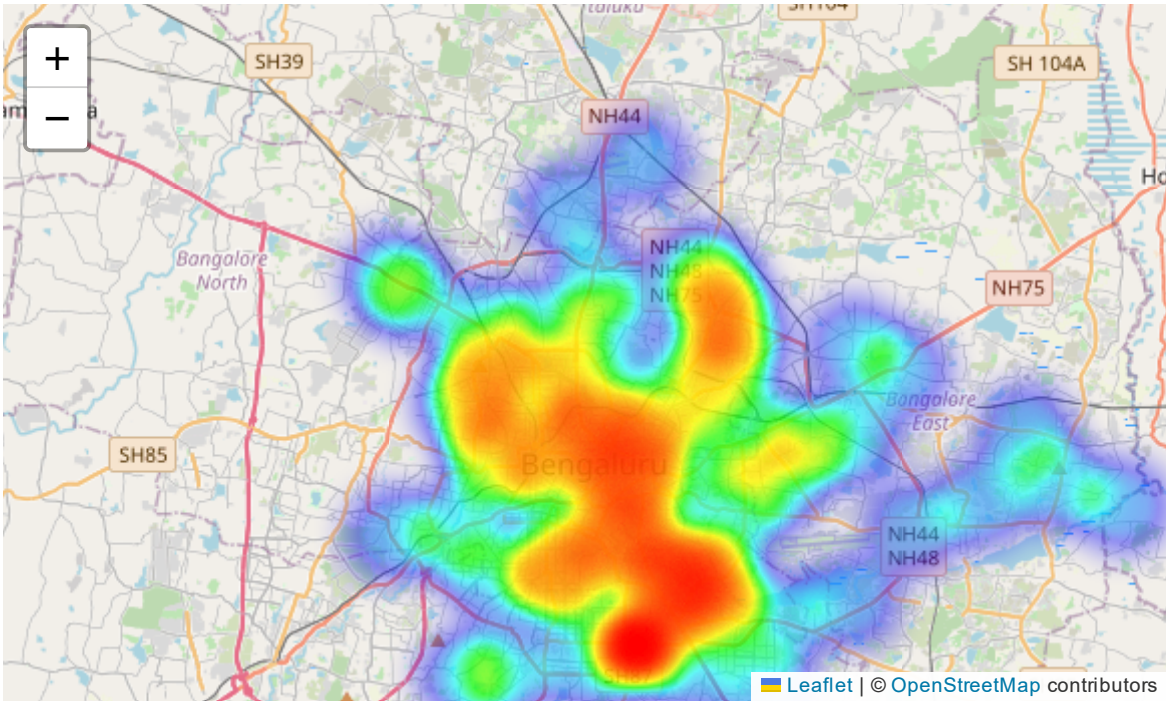
Out[195]:



In [196]:
```
#for North Indian cuisines
get_heatmap('North Indian')
```

```
                                                Area  count   latitude  lon
gitude
0           BTM , Bangalore  , Karnataka , India    2469  12.916360  77.
604733
1           HSR , Bangalore  , Karnataka , India    1123  12.900563  77.
649475
2    Whitefield , Bangalore  , Karnataka , India    1059  12.969637  77.
749745
3  Marathahalli , Bangalore  , Karnataka , India    1038  12.955257  77.
698416
```

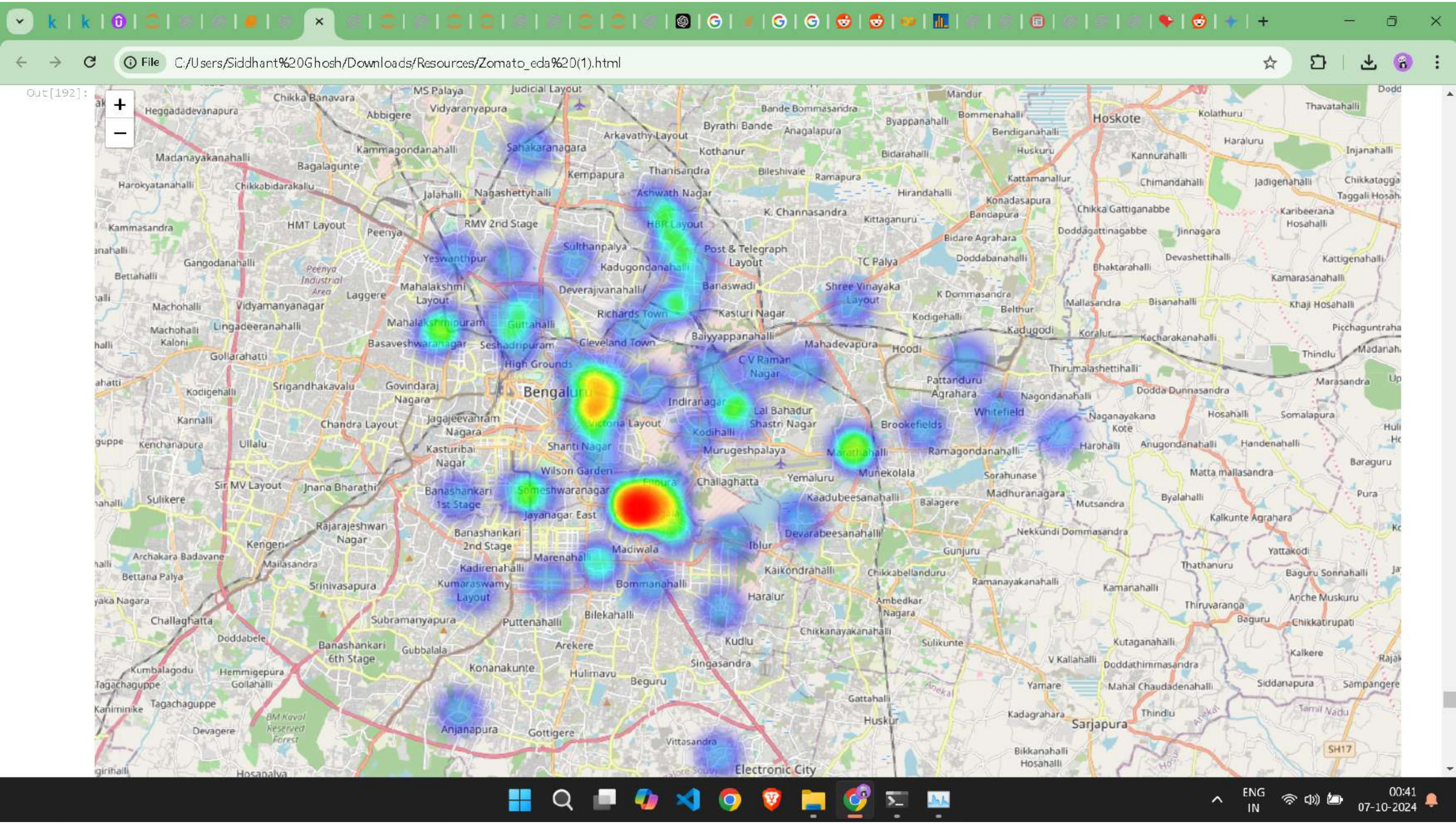
Out[196]:



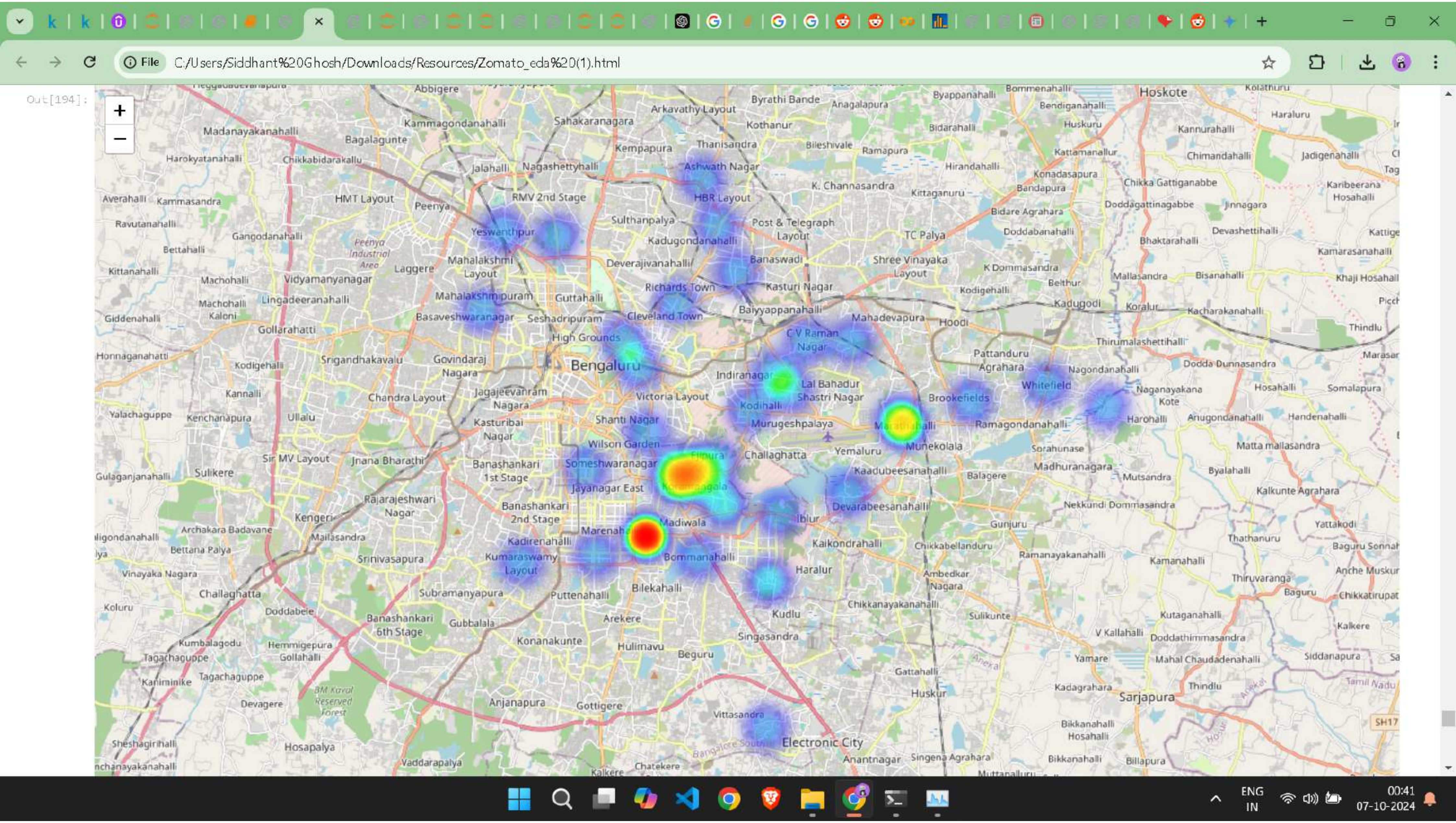
Conclusion

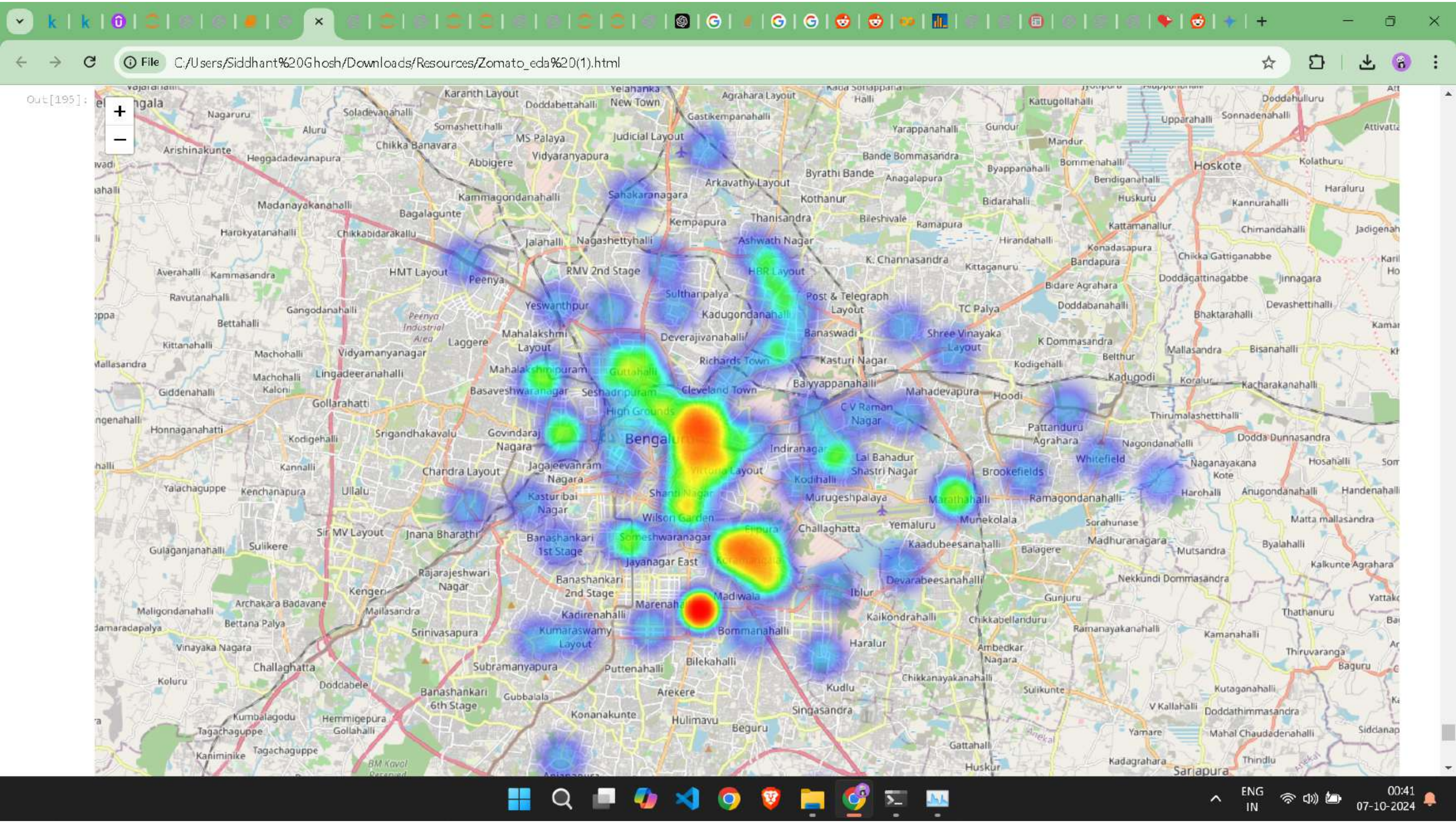
- In this exploratory data analysis of the Zomato dataset, we uncovered valuable insights into restaurant dynamics in Bangalore. Our analysis revealed a strong correlation between restaurant ratings and the acceptance of online orders, indicating that restaurants with higher ratings tend to receive more online orders.
- Furthermore, through unigram and bigram analyses, we identified prevalent food preferences among customers, with items such as Fried Rice, Paneer Butter Masala, Vanilla Icecream and Chicken Biryani frequently mentioned. This suggests that certain cuisines are particularly popular in the city, reflecting local tastes and dining habits.
- Geospatial analysis highlighted the concentration of restaurants in central Bangalore, with a notable decrease in density as one moves outward. This information can be instrumental for potential restaurant entrepreneurs seeking to establish their businesses in high-traffic areas.
- Overall, this EDA has provided a comprehensive understanding of customer preferences, restaurant performance, and geographical trends, laying a solid foundation for further research and business strategy development in the food and beverage sector.

In [ ]:



C:/Users/Siddhant%20Ghosh/Downloads/Resources/Zomato_eda%20(1).html
Out[192]:
Bengaluru
Hoskote
Sahakaranagara
Yeswanthpur
HBR Layout
Banaswadi
Mahalakshmipuram
Seshadripuram
Indiranagar
Koramangala
Whitefield
Marathahalli
Brookefields
Jayanagar East
Banashankari 1st Stage
Madiwala
Bommanahalli
Electronic City
Sarjapura
Anjanapura
Kumaraswamy Layout
Devarabeesanahalli
ENG
IN
00:41
07-10-2024


C:/Users/Siddhant%20Ghosh/Downloads/Resources/Zomato_eda%20(1).html
Out[194]:
Bengaluru
Yeswanthpur
Ashwath Nagar
HBR Layout
Banaswadi
Richards Town
Cleveland Town
Indiranagar
Lal Bahadur Shastri Nagar
Kodihalli
C V Raman Nagar
Brookefields
Whitefield
Marathahalli
Koramangala
Madiwala
Bommanahalli
Jayanagar East
Kumaraswamy Layout
Electronic City
Hoskote
Sarjapura
ENG
IN
00:41
07-10-2024



C:/Users/Siddhant%20Ghosh/Downloads/Resources/Zomato_eda%20(1).html
Out[195]:
Bengaluru
Yelahanka New Town
Judicial Layout
Sahakaranagara
Kempapura
Ashwath Nagar
HBR Layout
Yeswanthpur
Peenya
Sulthanpalya
Kadugondanahalli
Banaswadi
Richards Town
Cleveland Town
Mahalakshmipuram
Basaveshwaranagar
Seshadripuram
High Grounds
Indiranagar
C V Raman Nagar
Shree Vinayaka Layout
Mahadevapura
Whitefield
Brookefields
Marathahalli
Kodihalli
Murugeshpalya
Jagajeevanram Nagara
Kasturibai Nagar
Shanti Nagar
Wilson Garden
Someshwaranagar
Jayanagar East
Banashankari 1st Stage
Banashankari 2nd Stage
Madiwala
Bommanahalli
Kumaraswamy Layout
Kadirenahalli
Bilekahalli
Devarabeesanahalli
Kaadubeesanahalli
Iblur
Haralur
Kudlu
Singasandra
Hoskote
Sarjapura
ENG
IN
00:41
07-10-2024

Out[196]: