Data Analysis for Uber Eats

```python
# Importing Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from wordcloud import STOPWORDS
import folium
from folium import plugins
```

```python
# Loading Dataset
restaurants_menu  = pd.read_csv("restaurant-menus.csv")
restaurants = pd.read_csv("restaurants.csv")
```

```python
restaurants_menu.head()
```

	restaurant_id	category	name	description	price
0	1	Extra Large Pizza	Extra Large Meat Lovers	Whole pie.	15.99 USD
1	1	Extra Large Pizza	Extra Large Supreme	Whole pie.	15.99 USD
2	1	Extra Large Pizza	Extra Large Pepperoni	Whole pie.	14.99 USD
3	1	Extra Large Pizza	Extra Large BBQ Chicken & Bacon	Whole Pie	15.99 USD
4	1	Extra Large Pizza	Extra Large 5 Cheese	Whole pie.	14.99 USD

```python
restaurants.head()
```

	id	position	name	score	ratings	category	price_range	full_address	zip_code	lat	lng
0	1	19	PJ Fresh (224 Daniel Payne Drive)	NaN	NaN	Burgers, American, Sandwiches	$	224 Daniel Payne Drive, Birmingham, AL, 35207	35207	33.562365	-86.830703
1	2	9	J' ti`'z Smoothie-N-Coffee Bar	NaN	NaN	Coffee and Tea, Breakfast and Brunch, Bubble Tea	NaN	1521 Pinson Valley Parkway, Birmingham, AL, 35217	35217	33.583640	-86.773330
2	3	6	Philly Fresh Cheesesteaks (541-B Graymont Ave)	NaN	NaN	American, Cheesesteak, Sandwiches, Alcohol	$	541-B Graymont Ave, Birmingham, AL, 35204	35204	33.509800	-86.854640
3	4	17	Papa Murphy's (1580 Montgomery Highway)	NaN	NaN	Pizza	$	1580 Montgomery Highway, Hoover, AL, 35226	35226	33.404439	-86.806614
4	5	162	Nelson Brothers Cafe (17th St N)	4.7	22.0	Breakfast and Brunch, Burgers, Sandwiches	NaN	314 17th St N, Birmingham, AL, 35203	35203	33.514730	-86.811700

Checking Null Values

```
restaurants.isnull().sum()
```

```
id                0
position          0
name              0
score          2831
ratings        2831
category          1
price_range     581
full_address     22
zip_code         22
lat               0
lng               0
dtype: int64
```

```
restaurants_menu.isnull().sum()
```

```
restaurant_id          0
category               0
name                   3
description      1452140
price                  0
dtype: int64
```

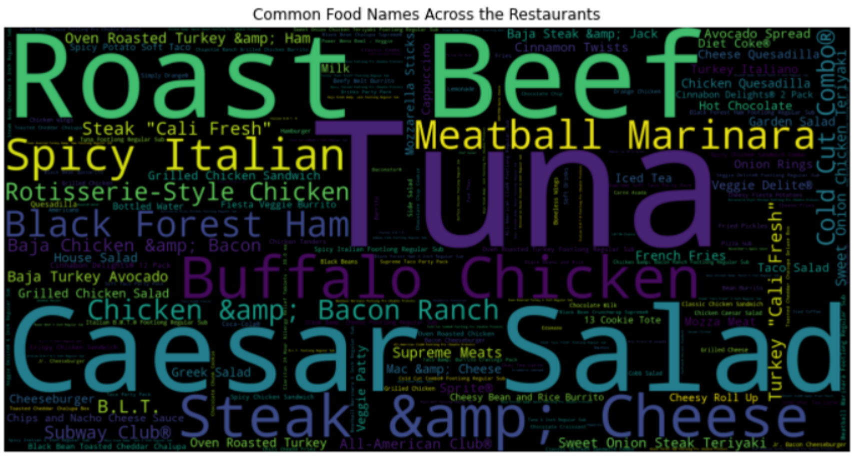
```
# Dropping the null values
restaurants.dropna(inplace=True)
restaurants_menu.dropna(inplace=True)
```

Analysis

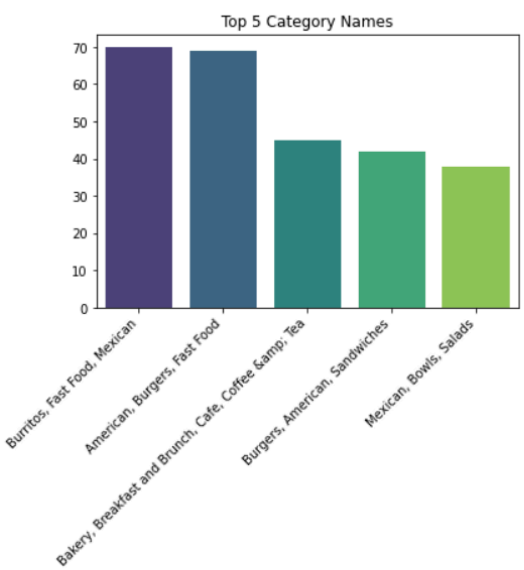
```
# Common food category across the restaurants
plt.figure(figsize=(12,18))
text = restaurants['category'].value_counts().to_dict()
stopwords = set(STOPWORDS)
wc =WordCloud(width = 2000, height = 1000, random_state = 1).generate_from_frequencies(text)
plt.imshow(wc)
plt.title("Common Food Category Across the Restaurants")
plt.axis('off')
plt.show()
```

Common Food Category Across the Restaurants

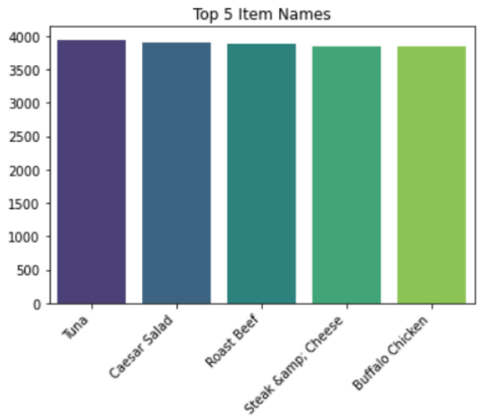
```python
# Common food name across the restaurants
plt.figure(figsize=(12,18))
data = restaurants_menu['name'].value_counts().to_dict()
stopwords = set(STOPWORDS)
wc =WordCloud(width = 2000, height = 1000, random_state = 1).generate_from_frequencies(data)
plt.imshow(wc)
plt.title("Common Food Names Across the Restaurants")
plt.axis('off')
plt.show()
```




```python
# Top 5 Category Food
catcount=restaurants['category'].value_counts().head(5)
sns.barplot(x=catcount.index, y=catcount.values, palette='viridis')
plt.title("Top 5 Category Names")
plt.xticks(rotation=45, ha='right')
plt.show()
```

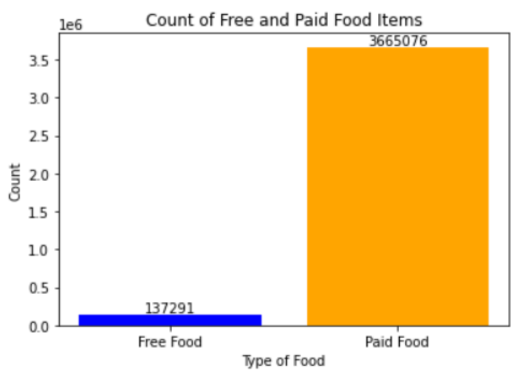


```
# Top 5 Item names
namecount=restaurants_menu['name'].value_counts().head(5)
sns.barplot(x=namecount.index, y=namecount.values, palette='viridis')
plt.title("Top 5 Item Names")
plt.xticks(rotation = 45, ha = 'right')
plt.show()
```




```
# Dividing data in the free food and paid food in two categories
free_food = restaurants_menu[restaurants_menu.price=='0.0 USD']
paid_food = restaurants_menu[restaurants_menu!='0.0 USD']
```

```
counts = [len(free_food),len(paid_food)]
labels = ['Free Food','Paid Food']

fig, ax = plt.subplots()
bars = ax.bar(labels, counts, color=['blue', 'orange'])

# Add data labels
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x()+bar.get_width()/2, yval, yval, ha='center', va='bottom')

plt.title('Count of Free and Paid Food Items')
plt.xlabel('Type of Food')
plt.ylabel('Count')
plt.show()
```



```
# Finding outlets and chains
outlets = restaurants["name"].value_counts()
outlets.head()
```

```
Everyday Needs by Gopuff          4
Mad Chicken                       2
Bayleaf Authentic Indian Cuisine  2
Mad Chicken (Green Bay)           2
Starbucks (6th &amp; Drexel)      1
Name: name, dtype: int64
```

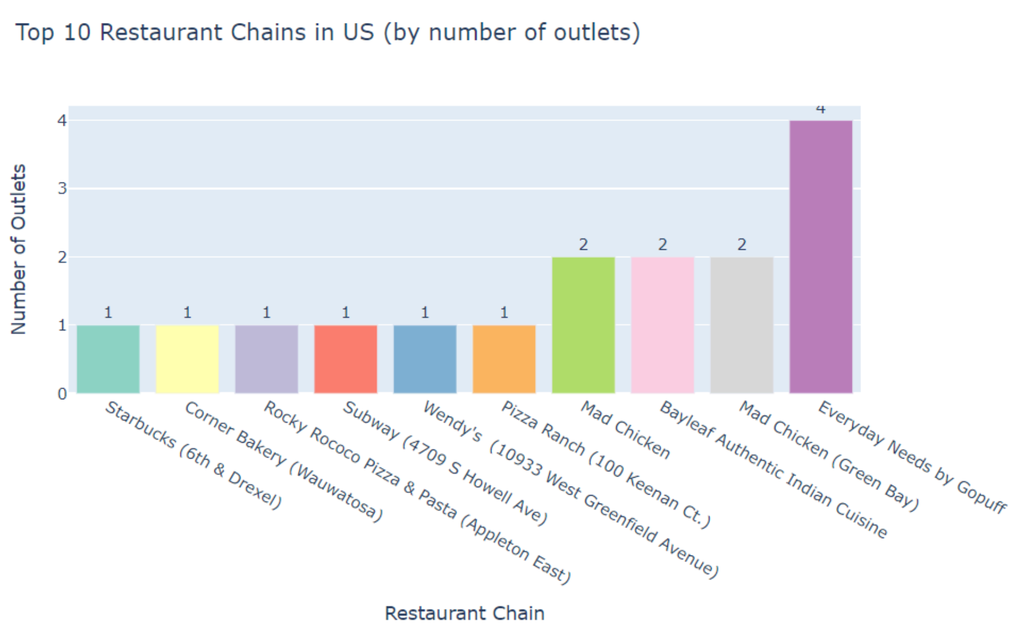
```
chains = outlets[outlets >= 2]
single = outlets[outlets == 1]
```

```
print("Total Restaurants = ", restaurants.shape[0])
print("Total Restaurants that are part of some chain = ", restaurants.shape[0] - single.shape[0])
print("Percentage of Restaurants that are part of a chain = ", np.round((restaurants.shape[0] - single.shape[0])
                                                                        / restaurants.shape[0],2)*100, "%")
```

```
Total Restaurants =  1898
Total Restaurants that are part of some chain =  10
Percentage of Restaurants that are part of a chain =  1.0 %
```

```
top10_chains = restaurants["name"].value_counts()[:10].sort_values(ascending=True)
```

```
# Finding top 10 chains in US by their outlet numbers
import plotly.express as px
fig = px.bar(top10_chains, orientation='v', color=top10_chains.index, color_discrete_sequence=px.colors.qualitative.Set3,
             labels={'index': 'Restaurant Chain', 'value': 'Number of Outlets'},
             title='Top 10 Restaurant Chains in US (by number of outlets)',
             width=800, height=500)

fig.update_layout(showlegend=False)  # Hide legend

# Displaying count values on top of each bar
fig.update_traces(texttemplate='%{value}', textposition='outside')

fig.show()
```



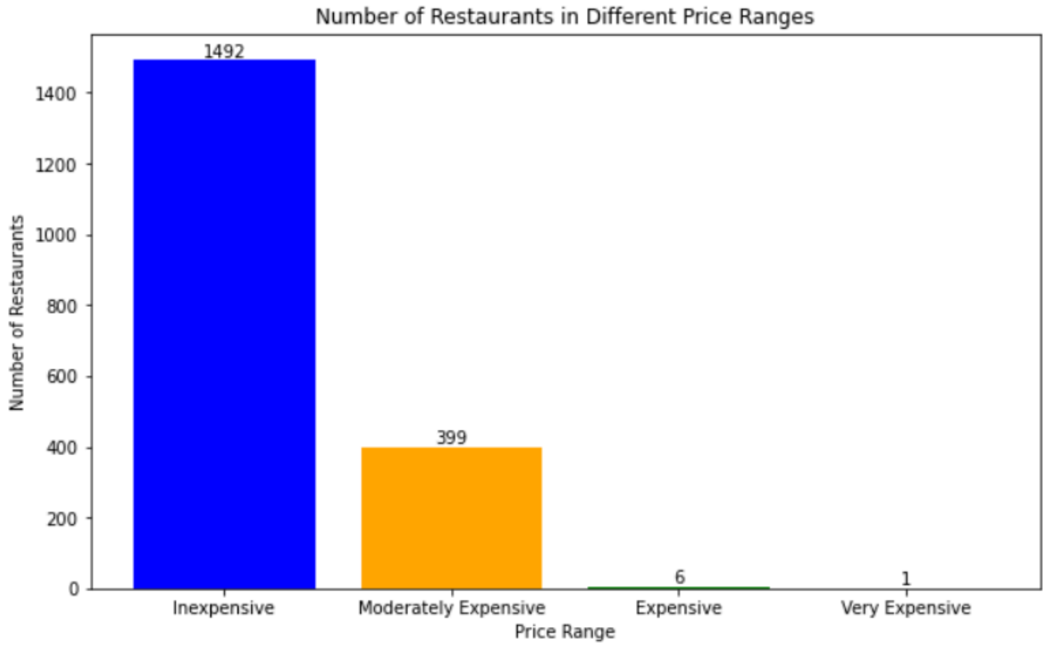
```python
mergedf = pd.merge(restaurants,restaurants_menu,left_on='id',right_on='restaurant_id')
mergedf.head()
```

	id	position	name_x	score	ratings	category_x	price_range	full_address	zip_code	lat	lng	restaurant_id	category_y	name_y	description
0	7	27	Jinsei Sushi	4.7	63.0	Sushi, Asian, Japanese	$	1830 29th Ave S, Birmingham, AL, 35209	35209	33.48044	-86.79044	7	Picked for you	Red Dragon	Spicy tuna tartare, tuna tataki, cucumber, sca...
1	7	27	Jinsei Sushi	4.7	63.0	Sushi, Asian, Japanese	$	1830 29th Ave S, Birmingham, AL, 35209	35209	33.48044	-86.79044	7	Picked for you	Edamame	Steamed or grilled. Served with ponzu.
2	7	27	Jinsei Sushi	4.7	63.0	Sushi, Asian, Japanese	$	1830 29th Ave S, Birmingham, AL, 35209	35209	33.48044	-86.79044	7	Picked for you	Spicy Hotate	Spicy scallop, avocado, crab salad, tobiko, te...
3	7	27	Jinsei Sushi	4.7	63.0	Sushi, Asian, Japanese	$	1830 29th Ave S, Birmingham, AL, 35209	35209	33.48044	-86.79044	7	Picked for you	Kadoma Tuna	Spicy tuna tartare, tempura rice cake, avocado...
4	7	27	Jinsei Sushi	4.7	63.0	Sushi, Asian, Japanese	$	1830 29th Ave S, Birmingham, AL, 35209	35209	33.48044	-86.79044	7	Picked for you	Spiro Roll	Spicy tuna tartare, sake, tempura asparagus, a...

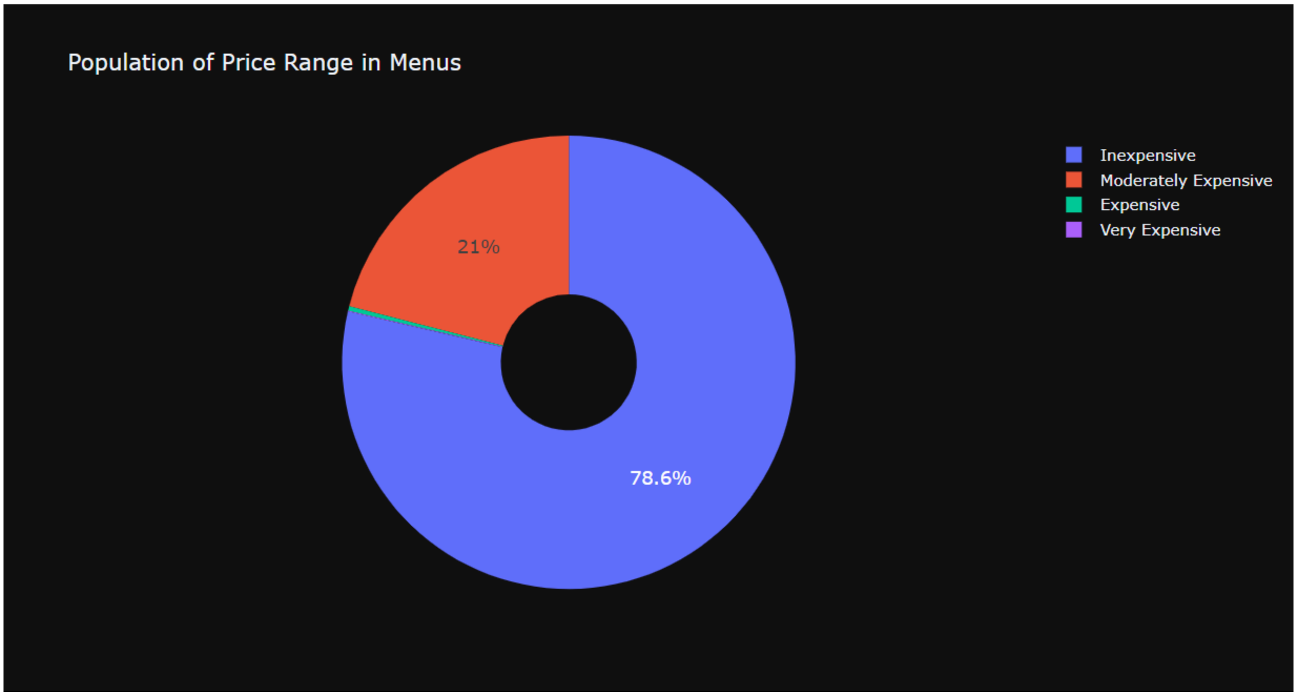
```python
plt.figure(figsize=(10,6))
a = len(restaurants[restaurants.price_range == '$'])
b = len(restaurants[restaurants.price_range == '$$'])
c =len(restaurants[restaurants.price_range == '$$$'])
d = len(restaurants[restaurants.price_range == '$$$$'])
print('$ = Inexpensive:',a)
print('$$ = Moderately Expensive:',b )
print('$$$ = Expensive:',c)
print('$$$$ = Very Expensive:',d)
```

```
$ = Inexpensive: 1492
$$ = Moderately Expensive: 399
$$$ = Expensive: 6
$$$$ = Very Expensive: 1

<Figure size 720x432 with 0 Axes>
```

```python
categories = ['Inexpensive', 'Moderately Expensive', 'Expensive', 'Very Expensive']
counts = [a, b, c, d]

# Increase the figure size
plt.figure(figsize=(10, 6))

# Create a bar plot
bars = plt.bar(categories, counts, color=['blue', 'orange', 'green', 'red'])
plt.xlabel('Price Range')
plt.ylabel('Number of Restaurants')
plt.title('Number of Restaurants in Different Price Ranges')

# Display data labels on top of each bar
for bar, count in zip(bars, counts):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), str(count),
             ha='center', va='bottom', color='black', fontsize=10)

plt.show()
```




```python
import plotly.express as px
fig = px.pie(restaurants, names='price_range',title='Population of Price Range in Menus', template = 'plotly_dark', hole = 0.3)
fig.update_traces(textposition='inside')
fig.update_layout(uniformtext_minsize=14, uniformtext_mode='hide',showlegend=True)

def newLegend(fig, newNames):
    for item in newNames:
        for i, elem in enumerate(fig.data[0].labels):
            if elem == item:
                fig.data[0].labels[i] = newNames[item]
    return(fig)

fig = newLegend(fig = fig, newNames = {'$':'Inexpensive',
                                        '$$' : 'Moderately Expensive',
                                       '$$$' : 'Expensive',
                                       '$$$$' : 'Very Expensive'})


fig.show()
```

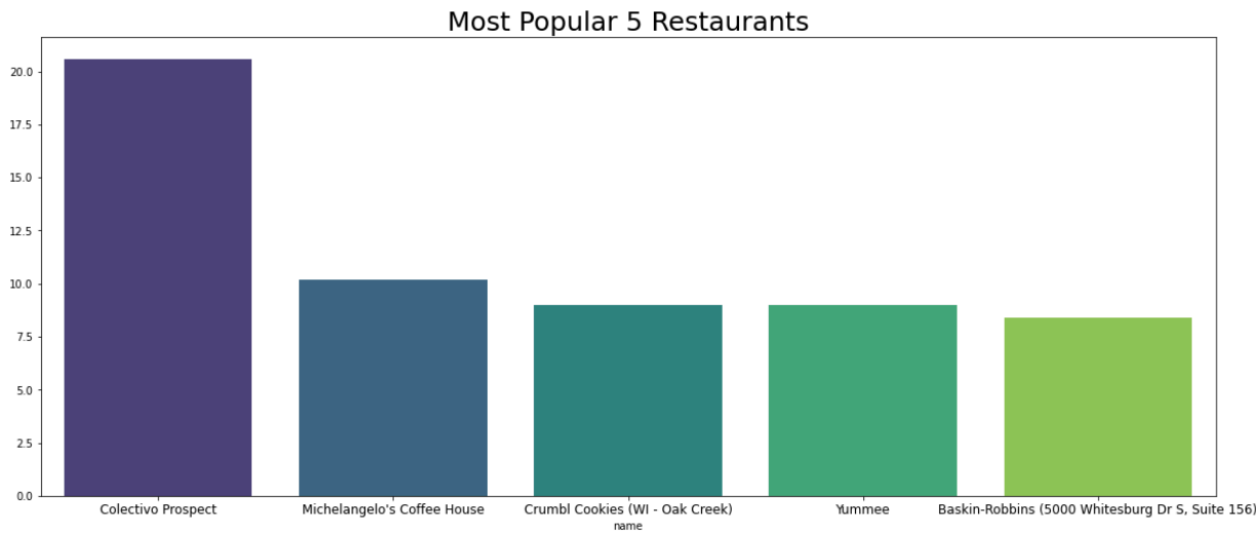



```python
popular_restaurant = restaurants.sort_values(['score','ratings'], ascending=False)
popular_restaurant.head()
```

	id	position	name	score	ratings	category	price_range	full_address	zip_code	lat	lng
2105	2106	188	Colectivo Prospect	5.0	103.0	Coffee and Tea, American, Breakfast and Brunch	$	2211 North Prospect Avenue, Milwaukee, WI, 53202	53202	43.059145	-87.885167
3096	3097	25	Michelangelo's Coffee House	5.0	51.0	Coffee and Tea, Bakery, Juice and Smoothies, A...	$	114 State Street, Madison, WI, 53703	53703	43.075016	-89.387200
2511	2512	12	Crumbl Cookies (WI - Oak Creek)	5.0	45.0	Desserts, Coffee and Tea	$	160 w town square way, oak creek, WI, 53154	53154	42.900540	-87.914760
3140	3141	23	Yummee	5.0	45.0	American, Desserts, Coffee and Tea	$$	5510 University Avenue, Madison, WI, 53705	53705	43.083090	-89.475560
964	965	7	Baskin-Robbins (5000 Whitesburg Dr S, Suite 156)	5.0	42.0	Desserts, Ice Cream + Frozen Yogurt, Comfort Food	$	5000 Whitesburg Dr SW, Huntsville, AL, 35802	35802	34.690520	-86.569150

```python
fig = plt.figure(figsize = (20,8))
ax =sns.barplot(popular_restaurant['name'][0:5],(popular_restaurant['ratings']/popular_restaurant['score'])
                [0:5],palette="viridis")
ax.set_xticklabels(ax.get_xticklabels(),fontsize = 12)
ax.set_title("Most Popular 5 Restaurants", fontsize =25)
```

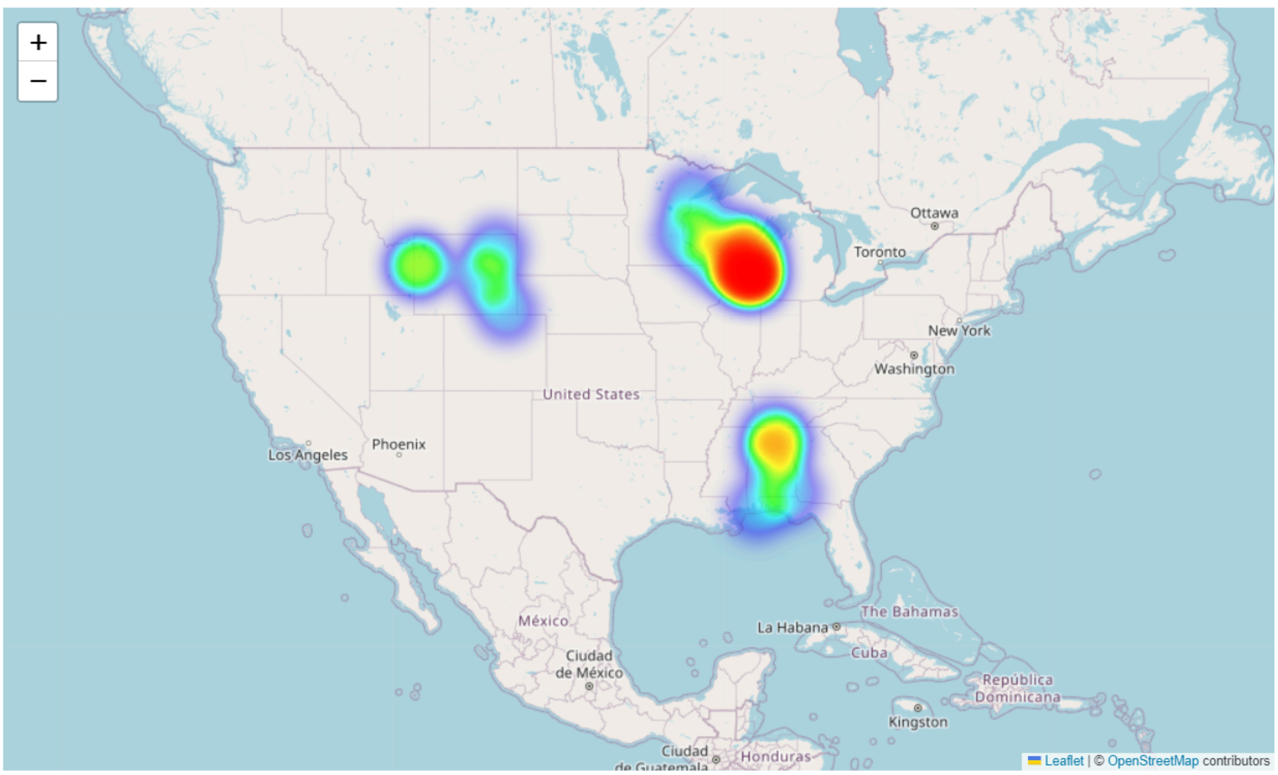
```
Text(0.5, 1.0, 'Most Popular 5 Restaurants')
```



Heatmap showing the restaurants in the US:

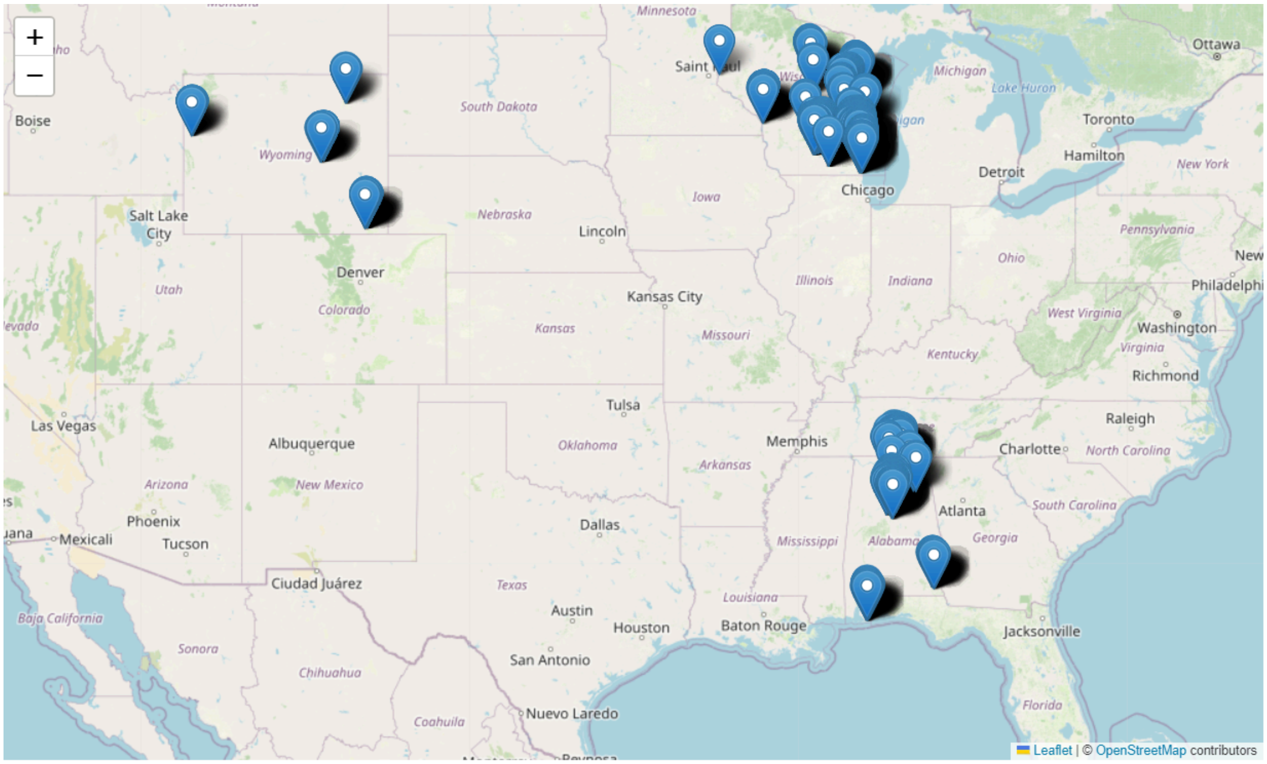
```
df_new = popular_restaurant.dropna()
```

```
# Extracting latitude and longitude
df_new = df_new[(df_new.lat.notnull())]
df_new = df_new[(df_new.lat != -1) & (df_new.lng != -1)]
df_new = df_new[~df_new.lat.isna()]

# Create a list of latitude and longitude pairs
locations = df_new[['lat', 'lng']].values.tolist()

# Create a Folium map centered at the specified location
map_offenses = folium.Map(location=[37.09024, -95.712891], zoom_start=4.4)

# Add a heatmap layer to the map using the locations
plugins.HeatMap(locations).add_to(map_offenses)

# Display the map
map_offenses
```



Folium Markers showing all the restuarants:

```
map_offenses = folium.Map(location=[37.09024,-95.712891], zoom_start=4.4)

for i, loc in df_new.iterrows():
    folium.Marker((loc['lat'], loc['lng'])).add_to(map_offenses)
map_offenses
```



Created MarkerCluster using Folium:

```
fivestar = df_new[df_new['score']==5]
dffs = pd.DataFrame(fivestar)
```

```
import folium
from folium.plugins import MarkerCluster

# Make an empty map
m = folium.Map(location=[37.09024, -95.712891], tiles="OpenStreetMap", zoom_start=4.2)

# Create a MarkerCluster layer
marker_cluster = MarkerCluster().add_to(m)

# Add marker one by one on the map using MarkerCluster
for i, loc in fivestar.iterrows():
    folium.Marker(
        (loc['lat'], loc['lng']),
        icon=folium.Icon(color="red", icon="glyphicon-cutlery"),
    ).add_to(marker_cluster)

# Show the map again
m
```