online restaurant rating

importiing libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#import plotly.plotly as py
#import plotly.graph_objs as go

#py.offline.init_notebook_mode(connected=True)

#%matplotlib notebook
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')
```

In [6]:
```
# import data set
data = pd.read_csv("C:/Users/MONIKA/Downloads/Untitled Folder/zomato.csv")
data
```

Out[6]:

	url	address	name	online_order	book_table	rate	votes	
0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1/5	775	42
1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1/5	787	
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8/5	918	+9
3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7/5	88	+9
4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8/5	166	8026
...	...	...	...	...	...	...	...	...
51712	https://www.zomato.com/bangalore/best-brews-fo...	Four Points by Sheraton Bengaluru, 43/3, White...	Best Brews - Four Points by Sheraton Bengaluru...	No	No	3.6/5	27	
51713	https://www.zomato.com/bangalore/vinod-bar-and...	Number 10, Garudachar Palya, Mahadevapura, Whi...	Vinod Bar And Restaurant	No	No	NaN	0	+9
51714	https://www.zomato.com/bangalore/plunge-sherat...	Sheraton Grand Bengaluru Whitefield Hotel & Co...	Plunge - Sheraton Grand Bengaluru Whitefield H...	No	No	NaN	0	
51715	https://www.zomato.com/bangalore/chime-sherato...	Sheraton Grand Bengaluru Whitefield Hotel & Co...	Chime - Sheraton Grand Bengaluru Whitefield Ho...	No	Yes	4.3/5	236	
51716	https://www.zomato.com/bangalore/the-nest-the-...	ITPL Main Road, KIADB Export Promotion Industr...	The Nest - The Den Bengaluru	No	No	3.4/5	13	+

51717 rows × 17 columns

data preprocessing

In [5]:
```
data.isna().sum()
```

Out[5]:
```
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 [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51717 entries, 0 to 51716
Data columns (total 17 columns):
url                            51717 non-null object
address                        51717 non-null object
name                           51717 non-null object
online_order                   51717 non-null object
book_table                     51717 non-null object
rate                           43942 non-null object
votes                          51717 non-null int64
phone                          50509 non-null object
location                       51696 non-null object
rest_type                      51490 non-null object
dish_liked                     23639 non-null object
cuisines                       51672 non-null object
approx_cost(for two people)    51371 non-null object
reviews_list                   51717 non-null object
menu_item                      51717 non-null object
listed_in(type)                51717 non-null object
listed_in(city)                51717 non-null object
dtypes: int64(1), object(16)
memory usage: 6.7+ MB
```

In [8]:
```
data=data[data.cuisines.isna()==False]
```

In [9]:
```
data.isna().sum()
```

Out[9]:
```
url                            0
address                        0
name                           0
online_order                   0
book_table                     0
rate                        7741
votes                          0
phone                       1179
location                       0
rest_type                    206
dish_liked                 28033
cuisines                       0
approx_cost(for two people)  320
reviews_list                   0
menu_item                      0
listed_in(type)                0
listed_in(city)                0
dtype: int64
```

In [10]:
```
data.isna().sum()
```

Out[10]:
```
url                            0
address                        0
name                           0
online_order                   0
book_table                     0
rate                        7741
votes                          0
phone                       1179
location                       0
rest_type                    206
dish_liked                 28033
cuisines                       0
approx_cost(for two people)  320
reviews_list                   0
menu_item                      0
listed_in(type)                0
listed_in(city)                0
dtype: int64
```

In [11]:
```
data.drop(columns=["url", 'address','phone','listed_in(city)'], inplace  =True)
```

In [12]:
```
data.rename(columns={'approx_cost(for two people)': 'average_cost'}, inplace=True)
```

In [13]:
```
data.rename(columns={'listed_in(type)': 'listed_type'}, inplace=True)
```

In [14]:
```
data.name.value_counts().head()
```

Out[14]:
```
Cafe Coffee Day      96
Onesta               85
Just Bake            73
Empire Restaurant    71
Five Star Chicken    70
Name: name, dtype: int64
```

In [16]:
```
data.online_order.value_counts()
```

Out[16]:
```
Yes     30428
No      21244
Name: online_order, dtype: int64
```

data visualization

In [17]:
```
ax= sns.countplot(data['online_order'])
plt.title('Number of Restaurants accepting online orders', weight='bold')
plt.xlabel('online orders')
```

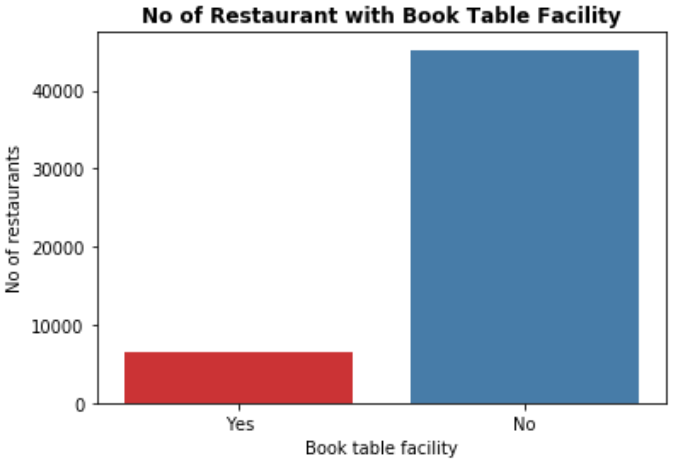
Out[17]:
```
Text(0.5, 0, 'online orders')
```

In [18]:
```
data['book_table'].value_counts()
```

Out[18]:
```
No      45223
Yes      6449
Name: book_table, dtype: int64
```

In [19]:
```
sns.countplot(data['book_table'], palette= "Set1")
plt.title("No of Restaurant with Book Table Facility", weight = 'bold')
plt.xlabel('Book table facility')
plt.ylabel('No of restaurants')
```

Out[19]:
```
Text(0, 0.5, 'No of restaurants')
```



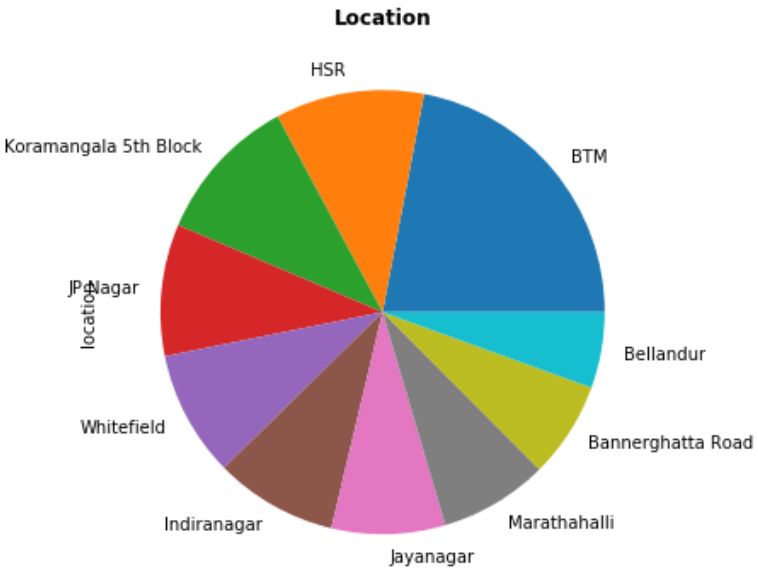
In [20]:
```
data['location'].value_counts()[:10]
```

Out[20]:
```
BTM                    5124
HSR                    2523
Koramangala 5th Block  2504
JP Nagar               2233
Whitefield             2136
Indiranagar            2081
Jayanagar              1926
Marathahalli           1843
Bannerghatta Road      1630
Bellandur              1286
Name: location, dtype: int64
```

Pie chart

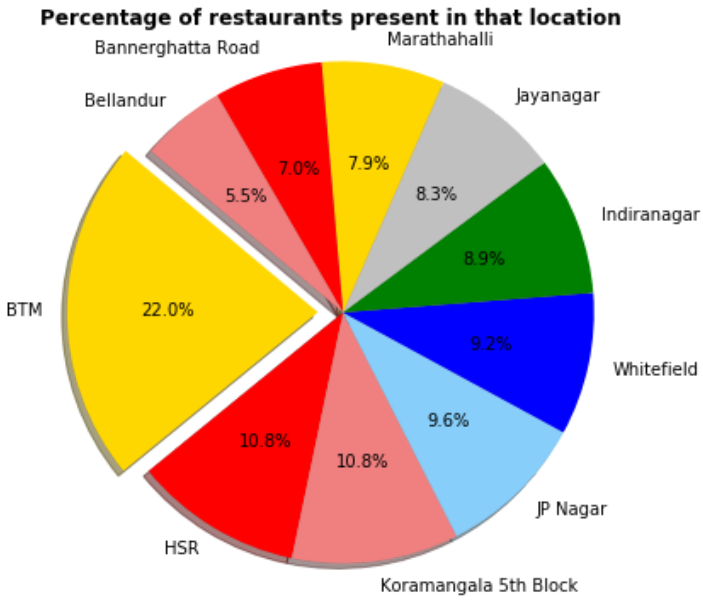
In [21]:
```
plt.figure(figsize=(12,6))
data['location'].value_counts()[:10].plot(kind = 'pie')
plt.title('Location', weight = 'bold')
```

Out[21]:
```
Text(0.5, 1.0, 'Location')
```



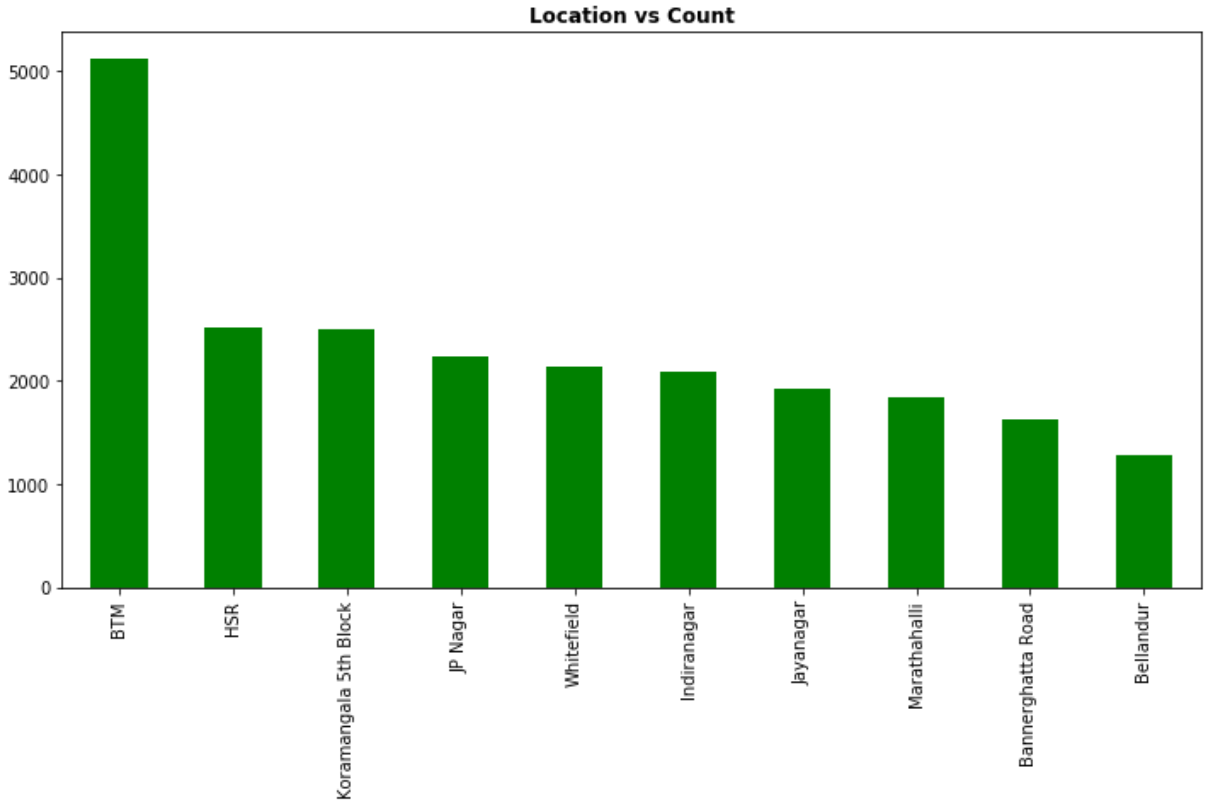
In [22]:

```
plt.figure(figsize = (12,6))
names = data['location'].value_counts()[:10].index
values = data['location'].value_counts()[:10].values
colors = ['gold', 'red', 'lightcoral', 'lightskyblue','blue','green','silver']
explode = (0.1, 0, 0, 0,0,0,0,0,0,0)  # explode 1st slice

plt.pie(values, explode=explode, labels=names, colors=colors,autopct='%1.1f%%', shadow=True, star
plt.axis('equal')
plt.title("Percentage of restaurants present in that location", weight = 'bold')
plt.show()
```



In [23]:
```
plt.figure(figsize = (12,6))
data['location'].value_counts()[:10].plot(kind = 'bar', color = 'g')
plt.title("Location vs Count", weight = 'bold')
```

Out[23]: Text(0.5, 1.0, 'Location vs Count')



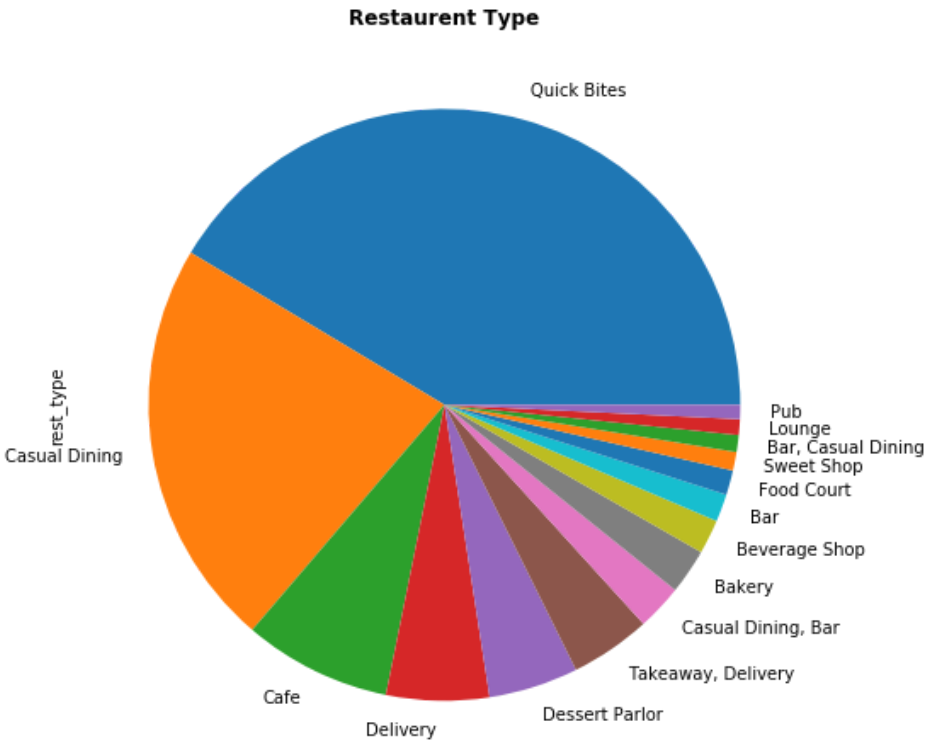
In [24]:
```
data['location'].nunique()
```

Out[24]: 93

In [25]:
```
data['rest_type'].value_counts().head(10)
```

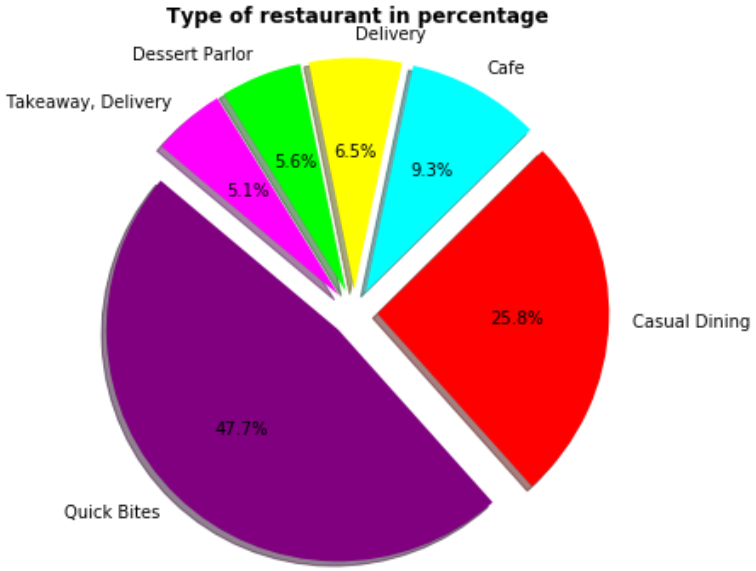
Out[25]:
```
Quick Bites           19129
Casual Dining         10326
Cafe                   3732
Delivery               2595
Dessert Parlor         2262
Takeaway, Delivery     2035
Casual Dining, Bar     1154
Bakery                 1141
Beverage Shop           865
Bar                     697
Name: rest_type, dtype: int64
```

In [26]:
```
plt.figure(figsize = (14,8))
data.rest_type.value_counts()[:15].plot(kind = 'pie')
plt.title('Restaurent Type', weight = 'bold')
plt.show()
```



In [27]:
```
colors = ['#800080','red','#00FFFF','#FFFF00','#00FF00','#FF00FF']
```

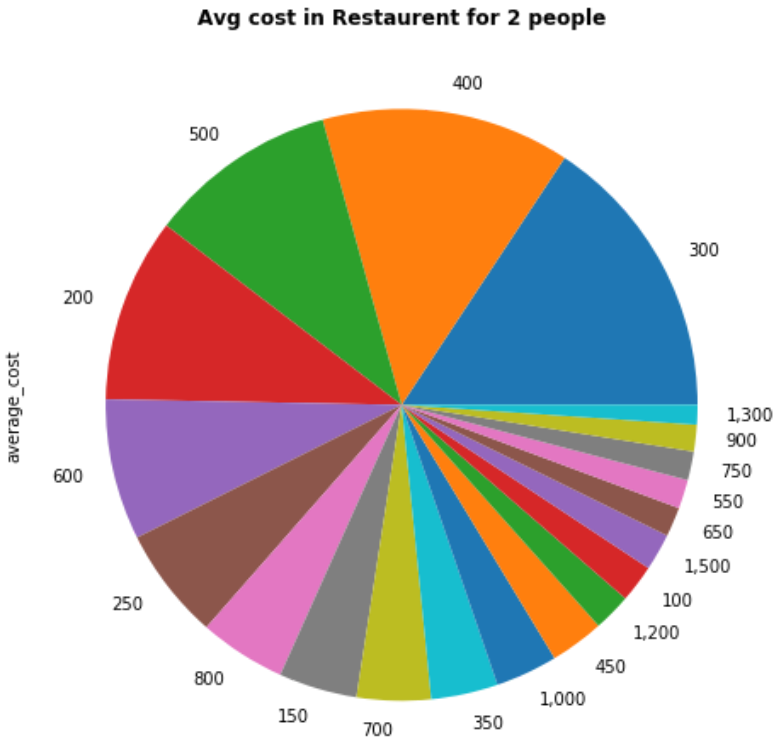
In [28]:
```
plt.figure(figsize = (12,6))
names = data['rest_type'].value_counts()[:6].index
values = data['rest_type'].value_counts()[:6].values
explode = (0.1, 0.1, 0.1, 0.1,0.1,0.1)  # explode 1st slice

plt.title('Type of restaurant in percentage', weight = 'bold')
plt.pie(values, explode=explode, labels=names, colors=colors,autopct='%1.1f%%', shadow=True, star
plt.axis('equal')
plt.show()
```



In [29]:
```
data['average_cost'].value_counts()[:20]
```

Out[29]:
```
300      7576
400      6554
500      4977
200      4855
600      3712
250      2959
800      2285
150      2064
700      1948
350      1763
1,000    1637
450      1417
1,200     993
100       991
1,500     971
650       776
550       761
750       758
900       700
1,300     516
Name: average_cost, dtype: int64
```

In [30]:
```
plt.figure(figsize = (12,8))
data['average_cost'].value_counts()[:20].plot(kind = 'pie')
plt.title('Avg cost in Restaurent for 2 people', weight = 'bold')
plt.show()
```



In [31]:
```
colors  = ("red", "green", "orange", "cyan", "brown", "grey", "blue", "indigo", "beige", "yellow"
```

In [33]:
```
dishes_data = data[data.dish_liked.notnull()]
dishes_data.dish_liked = dishes_data.dish_liked.apply(lambda x:x.lower().strip())
```

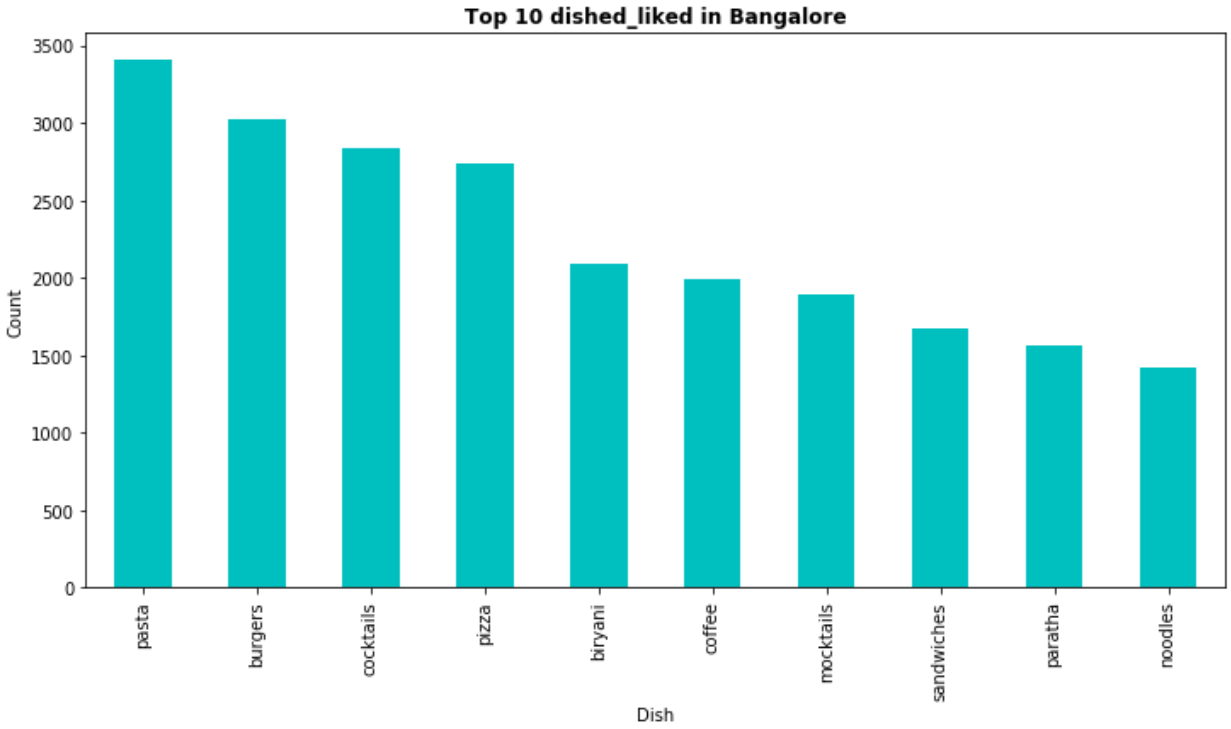
In [34]:
```
dishes_data.isnull().sum()
```

Out[34]:
```
name              0
online_order      0
book_table        0
rate             30
votes             0
location          0
rest_type        70
dish_liked        0
cuisines          0
average_cost    136
reviews_list      0
menu_item         0
listed_type       0
dtype: int64
```

In [35]:
```
# count each dish to see how many times each dish repeated
dish_count = []
for i in dishes_data.dish_liked:
    for t in i.split(','):
        t = t.strip() # remove the white spaces to get accurate results
        dish_count.append(t)
```

In [36]:
```
plt.figure(figsize=(12,6))
pd.Series(dish_count).value_counts()[:10].plot(kind='bar',color= 'c')
plt.title('Top 10 dished_liked in Bangalore',weight='bold')
plt.xlabel('Dish')
plt.ylabel('Count')
```

Out[36]:
```
Text(0, 0.5, 'Count')
```

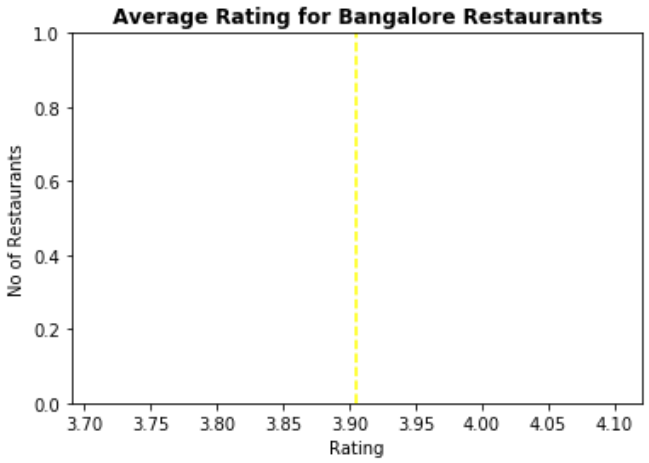


In [37]:
```
data['rate'] = data['rate'].replace('NEW',np.NaN)
data['rate'] = data['rate'].replace('-',np.NaN)
data.dropna(how = 'any', inplace = True)
```

In [38]:
```
data['rate'] = data.loc[:,'rate'].replace('[ ]','',regex = True)
data['rate'] = data['rate'].astype(str)
data['rate'] = data['rate'].apply(lambda r: r.replace('/5',''))
data['rate'] = data['rate'].apply(lambda r: float(r))
```

In [39]:
```
plt.axvline(x= data.rate.mean(),ls='--',color='yellow')
plt.title('Average Rating for Bangalore Restaurants',weight='bold')
plt.xlabel('Rating')
plt.ylabel('No of Restaurants')
print(data.rate.mean())
```

```
3.9058343007007914
```



In [40]:
```
data['online_order']= pd.get_dummies(data.online_order, drop_first=True)
data['book_table']= pd.get_dummies(data.book_table, drop_first=True)
data
```

Out[40]:

	name	online_order	book_table	rate	votes	location	rest_type	dish_liked	cuisines	average_cost
0	Jalsa	1	1	4.1	775	Banashankari	Casual Dining	Pasta, Lunch Buffet, Masala Papad, Paneer Laja...	North Indian, Mughlai, Chinese	800
1	Spice Elephant	1	0	4.1	787	Banashankari	Casual Dining	Momos, Lunch Buffet, Chocolate Nirvana, Thai G...	Chinese, North Indian, Thai	800
2	San Churro Cafe	1	0	3.8	918	Banashankari	Cafe, Casual Dining	Churros, Cannelloni, Minestrone Soup, Hot Choc...	Cafe, Mexican, Italian	800
3	Addhuri Udupi Bhojana	0	0	3.7	88	Banashankari	Quick Bites	Masala Dosa	South Indian, North Indian	300
4	Grand Village	0	0	3.8	166	Basavanagudi	Casual Dining	Panipuri, Gol Gappe	North Indian, Rajasthani	600
...	...	...	...	...	...	...	...	...	...	...
51705	Izakaya Gastro Pub	1	1	3.8	128	Whitefield	Bar, Casual Dining	Beer, Chicken Guntur, Paneer Tikka, Fish, Nood...	North Indian, Continental, Mediterranean	1,200
51707	M Bar - Bengaluru Marriott Hotel Whitefield	0	0	3.9	77	Whitefield	Fine Dining, Bar	Rooftop Ambience	Finger Food	2,000
51708	Keys Cafe - Keys Hotel	0	0	2.8	161	Whitefield	Casual Dining, Bar	Salads, Coffee, Breakfast Buffet, Halwa, Chick...	Chinese, Continental, North Indian	1,200
51711	Bhagini	0	0	2.5	81	Whitefield	Casual Dining, Bar	Biryani, Andhra Meal	Andhra, South Indian, Chinese, North Indian	800
51715	Chime - Sheraton Grand Bengaluru Whitefield Ho...	0	1	4.3	236	ITPL Main Road, Whitefield	Bar	Cocktails, Pizza, Buttermilk	Finger Food	2,500

23259 rows × 13 columns

In [41]:
```
data.drop(columns=['dish_liked','reviews_list','menu_item','listed_type'], inplace  =True)
```

In [42]:
```
data['rest_type'] = data['rest_type'].str.replace(',' , '')
data['rest_type'] = data['rest_type'].astype(str).apply(lambda x: ' '.join(sorted(x.split())))
data['rest_type'].value_counts().head()
```

Out[42]:
```
Casual Dining        7331
Bites Quick          5253
Cafe                 2375
Bar Casual Dining    1321
Dessert Parlor       1083
Name: rest_type, dtype: int64
```

In [43]:
```
data['rest_type'] = data['rest_type'].str.replace(',' , '')
data['rest_type'] = data['rest_type'].astype(str).apply(lambda x: ' '.join(sorted(x.split())))
data['rest_type'].value_counts().head()
```

Out[43]:
```
Casual Dining        7331
Bites Quick          5253
Cafe                 2375
Bar Casual Dining    1321
Dessert Parlor       1083
Name: rest_type, dtype: int64
```

In [45]:
```
data['cuisines'] = data['cuisines'].str.replace(',' , '')
data['cuisines'] = data['cuisines'].astype(str).apply(lambda x: ' '.join(sorted(x.split())))
data['cuisines'].value_counts().head()
```

Out[45]:
```
Indian North                          1152
Chinese Indian North                   852
Chinese Indian Indian North South      455
Indian South                           366
Cream Desserts Ice                     334
Name: cuisines, dtype: int64
```

label encoding

In [46]:
```
from sklearn.preprocessing import LabelEncoder
T = LabelEncoder()
data['location'] = T.fit_transform(data['location'])
data['rest_type'] = T.fit_transform(data['rest_type'])
data['cuisines'] = T.fit_transform(data['cuisines'])
#data['dish_liked'] = T.fit_transform(data['dish_liked'].
```

In [48]:
```
data["average_cost"] = data["average_cost"].str.replace(',' , '')
```

In [49]:
```
data["average_cost"] = data["average_cost"].astype('float')
```

In [50]:
```
data.head()
```

Out[50]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	average_cost
0	Jalsa	1	1	4.1	775	1	29	951	800.0
1	Spice Elephant	1	0	4.1	787	1	29	963	800.0
2	San Churro Cafe	1	0	3.8	918	1	22	806	800.0
3	Addhuri Udupi Bhojana	0	0	3.7	88	1	19	1201	300.0
4	Grand Village	0	0	3.8	166	4	29	1237	600.0

In [51]:
```
x = data.drop(['rate','name'],axis = 1)
```

In [52]:
```
y = data['rate']
```

In [53]:
```
x.shape
```

Out[53]:
```
(23259, 7)
```

In [54]:
```
y.shape
```

Out[54]:
```
(23259,)
```

dat training and testing

In [55]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 33)
```

In [56]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 23259 entries, 0 to 51715
Data columns (total 9 columns):
name            23259 non-null object
online_order    23259 non-null uint8
book_table      23259 non-null uint8
rate            23259 non-null float64
votes           23259 non-null int64
location        23259 non-null int32
rest_type       23259 non-null int32
cuisines        23259 non-null int32
average_cost    23259 non-null float64
dtypes: float64(2), int32(3), int64(1), object(1), uint8(2)
memory usage: 1.2+ MB
```

In [57]:
```
#standarizing
#taking numeric values
from sklearn.preprocessing import StandardScaler
num_values1=data.select_dtypes(['float64','int64']).columns
scaler = StandardScaler()
scaler.fit(data[num_values1])
data[num_values1]=scaler.transform(data[num_values1])
```

In [58]:
```
data.head()
```

Out[58]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	average_cost
0	Jalsa	1	1	0.455722	0.152328	1	29	951	0.089176
1	Spice Elephant	1	0	0.455722	0.163105	1	29	963	0.089176
2	San Churro Cafe	1	0	-0.248401	0.280757	1	22	806	0.089176
3	Addhuri Udupi Bhojana	0	0	-0.483109	-0.464668	1	19	1201	-0.871467
4	Grand Village	0	0	-0.248401	-0.394616	4	29	1237	-0.295081

linear regression

In [59]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train,y_train)
y_pred_lr = lr.predict(X_test)
```

In [60]:
```
lr.score(X_test, y_test)*100
```

Out[60]:
```
21.319197295401814
```

In [61]:
```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_lr))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred_lr))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred_lr)))
```

```
Mean Absolute Error: 0.2653603458127496
Mean Squared Error: 0.13773462897554362
Root Mean Squared Error: 0.3711261631514863
```

random forest

In [62]:
```
from sklearn import metrics
from sklearn.ensemble import RandomForestRegressor
rfr = RandomForestRegressor()
rfr.fit(X_train,y_train)
y_pred_rfr = rfr.predict(X_test)
```

In [63]:
```
rfr.score(X_test,y_test)*100
```

Out[63]:
```
90.99018454023702
```

In [64]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_rfr))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred_rfr))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred_rfr)))
```

```
Mean Absolute Error: 0.0466433200730223
Mean Squared Error: 0.015772126704752698
Root Mean Squared Error: 0.1255871279421291
```

logistic regression

In [70]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 33)
```

In [71]:
```
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(X_train,y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-71-a98b2fe02624> in <module>
      1 from sklearn.linear_model import LogisticRegression
      2 classifier=LogisticRegression()
----> 3 classifier.fit(X_train,y_train)

~\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py in fit(self, X, y, sample_weight)
   1531         X, y = check_X_y(X, y, accept_sparse='csr', dtype=_dtype, order="C",
   1532                          accept_large_sparse=solver != 'liblinear')
-> 1533         check_classification_targets(y)
   1534         self.classes_ = np.unique(y)
   1535         n_samples, n_features = X.shape

~\Anaconda3\lib\site-packages\sklearn\utils\multiclass.py in check_classification_targets(y)
    167     if y_type not in ['binary', 'multiclass', 'multiclass-multioutput',
    168                       'multilabel-indicator', 'multilabel-sequences']:
--> 169         raise ValueError("Unknown label type: %r" % y_type)
    170
    171

ValueError: Unknown label type: 'continuous'
```

In [ ]: