In [1]:

```
import numpy as np
import pandas as pd
import json
import itertools
import ast
import collections
import gc
```

In [2]:

```
# for reviews
from textblob import TextBlob
from nltk.sentiment.vader import SentimentIntensityAnalyzer
```

In [3]:

```
# Visualisation
import matplotlib as mpl
import matplotlib.pyplot as plt
#import matplotlib.pylab as pylab
import seaborn as sns
```

In [4]:

```
pd.set_option('display.max_rows',None)
pd.set_option('display.max_columns',None)
pd.set_option('max_colwidth', -1)
pd.set_option('display.float_format', '{:.2f}'.format)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\ipykernel_launcher.py:3: FutureWarning: Pass
ing a negative integer is deprecated in version 1.0 and will not be supported in future
version. Instead, use None to not limit the column width.
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [5]:

```
# Configure visualisations
%matplotlib inline
mpl.style.use( 'ggplot' )
plt.style.use('fivethirtyeight')
sns.set(context="notebook", palette="dark", style = 'whitegrid' , color_codes=True, rc=
params = {
    'axes.labelsize': "medium",
    'xtick.labelsize': 'medium',
    'legend.fontsize': 10,
    'figure.dpi': 100,
    'figure.figsize': [10, 7]
}
plt.rcParams.update(params)
```

Helper Functions

In [6]:

```
def strip_element(my_list):
    if(my_list!=None):
        return [x.strip() for x in my_list]
    else:
        return None
```

In [7]:

```
def count_element(my_list, word):
    if(my_list!=None):
        return [elem for elem in my_list].count(word)
    else:
        return None
```

In [8]:

```
def elem_type(val):
    x=str(type(val))
    x=x.replace("<","")
    x=x.replace(">","")
    x=x.replace("\'","")
    x=x.split()[1]
    return(x)
```

Read & Explore Business data

In [9]:

```
%%time
business1 = []
with open('./data/yelp_academic_dataset_business.json', 'r', encoding='utf-8') as f:
    for line in f:
        business1.append(json.loads(line))
print(business1[0])
```

```
{'business_id': 'f9NumwFMBDn751xgFiRbNA', 'name': 'The Range At Lake Norman', 'address':
'10913 Bailey Rd', 'city': 'Cornelius', 'state': 'NC', 'postal_code': '28031', 'latitud
e': 35.4627242, 'longitude': -80.8526119, 'stars': 3.5, 'review_count': 36, 'is_open':
1, 'attributes': {'BusinessAcceptsCreditCards': 'True', 'BikeParking': 'True', 'GoodForK
ids': 'False', 'BusinessParking': "{'garage': False, 'street': False, 'validated': Fals
e, 'lot': True, 'valet': False}", 'ByAppointmentOnly': 'False', 'RestaurantsPriceRange
2': '3'}, 'categories': 'Active Life, Gun/Rifle Ranges, Guns & Ammo, Shopping', 'hours':
{'Monday': '10:0-18:0', 'Tuesday': '11:0-20:0', 'Wednesday': '10:0-18:0', 'Thursday': '1
1:0-20:0', 'Friday': '11:0-20:0', 'Saturday': '11:0-20:0', 'Sunday': '13:0-18:0'}}
Wall time: 3.37 s
```

Convert json data to pandas dataframe

In [10]:

```
business=pd.DataFrame.from_dict(business1)
```

In [11]:

```
business.shape
```

Out[11]:

```
(209393, 14)
```

Extracting features from categories & Attribute columns

In [12]:

```
business["Is_Restaurant"]=None
```

Considered rows where 'Bar', 'Restaurants' & 'Food' exist in categories column as Restaurants.

Other similar words like 'pub', sandwitches ect mostly co-occur with above 3 words in categories and hence not considered

In [13]:

```
business.loc[(business["categories"].str.contains("Bar")==True) |
         (business["categories"].str.contains("Restaurants")==True) |
         (business["categories"].str.contains("Food")==True),"Is_Restaurant"]=True
```

In [14]:

```
business.loc[business["Is_Restaurant"]!=True, "Is_Restaurant"]=False
```

Filtering out rows which are restaurants only

In [15]:

```
restaurants=business[business["Is_Restaurant"]==True].reset_index(drop=True)
```

In [16]:

```
restaurants.shape
```

Out[16]:

```
(85798, 15)
```

In [17]:

```
del(business)
gc.collect()
```

Out[17]:

```
40
```

In [18]:

```
## top 20 restaurant names by review count
restaurants.groupby('name')['review_count'].count().reset_index().sort_values('review_c
```

Out[18]:

	name	review_count
46084	Starbucks	1182
31110	McDonald's	854
46595	Subway Restaurants	613
52491	Tim Hortons	388
7211	Burger King	337
47928	Taco Bell	334
38316	Pizza Hut	330
55366	Wendy's	323
55126	Walgreens	316
10842	Circle K	279
46593	Subway	243
14075	Domino's Pizza	232
14664	Dunkin'	223
24459	KFC	214
430	7-Eleven	209
10421	Chipotle Mexican Grill	191
23284	Jack in the Box	184

	name	review_count
36253	Panera Bread	166
23791	Jimmy John's	165
38861	Popeyes Louisiana Kitchen	159

In [19]:

```
# Since Subway Restaurants & Subway are same we'll rename 'Subway Restaurants' as 'Subw
restaurants.loc[restaurants["name"]=="Subway Restaurants", "name"]="Subway"
```

Checking if attributes column is not null

In [20]:

```
restaurants["attribute_exists"]=restaurants["attributes"].apply(lambda x : elem_type(x)
```

understanding attributes frequency to select the relevent features

In [21]:

```
%%time

attrib_list = []
for loop in range(len(business1)):
    if (business1[loop]['attributes']!=None):
        #print("loop==", loop)
        for key, vals in business1[loop]["attributes"].items():
            #print("loop==", loop)
            attrib_list.append(key.strip())
    else:
        k=1
        #print("Skipping")
```

```
Wall time: 421 ms
```

In [22]:

```
%%time
attrib_dict=collections.Counter(attrib_list)
attrib_dict=collections.OrderedDict(attrib_dict.most_common())
```

```
Wall time: 122 ms
```

In [23]:

```
pd.DataFrame.from_dict(attrib_dict, orient = 'index').reset_index().rename(columns = {
```

Out[23]:

	Attribute	Count
0	BusinessAcceptsCreditCards	122237
1	BusinessParking	115215
2	RestaurantsPriceRange2	111288
3	BikeParking	89765
4	GoodForKids	68535
5	RestaurantsTakeOut	66301
6	WiFi	65331

	Attribute	Count
7	ByAppointmentOnly	60799
8	OutdoorSeating	58441
9	RestaurantsDelivery	56679
10	RestaurantsGoodForGroups	56162
11	RestaurantsReservations	55361
12	Ambience	53806
13	HasTV	53388
14	Alcohol	50838
15	RestaurantsAttire	49567
16	NoiseLevel	46559
17	Caters	43969
18	GoodForMeal	35182
19	WheelchairAccessible	28635
20	RestaurantsTableService	20785
21	DogsAllowed	17539
22	BusinessAcceptsBitcoin	16532
23	HappyHour	15324
24	AcceptsInsurance	8660
25	Music	7912
26	BestNights	5483
27	GoodForDancing	5186
28	CoatCheck	4909
29	DriveThru	4282
30	Smoking	4270
31	BYOBCorkage	1425
32	HairSpecializesIn	1260
33	Corkage	1090
34	BYOB	740
35	AgesAllowed	136
36	DietaryRestrictions	61
37	Open24Hours	14
38	RestaurantsCounterService	13

In [24]:

```
del(business1)
```

```
gc.collect()
```

Out[24]:

```
40
```

Selecting Relevent Features from the above list based on counts

In [25]:

```
restaurants["RestaurantsPriceRange2"]=None
```

In [26]:

```
restaurants.loc[restaurants["attribute_exists"]=="dict","RestaurantsPriceRange2"] = [d.
```

In [27]:

```
restaurants.loc[restaurants["RestaurantsPriceRange2"]=="None","RestaurantsPriceRange2"]
```

In [28]:

```
restaurants.groupby('RestaurantsPriceRange2')["business_id"].count().reset_index()
```

Out[28]:

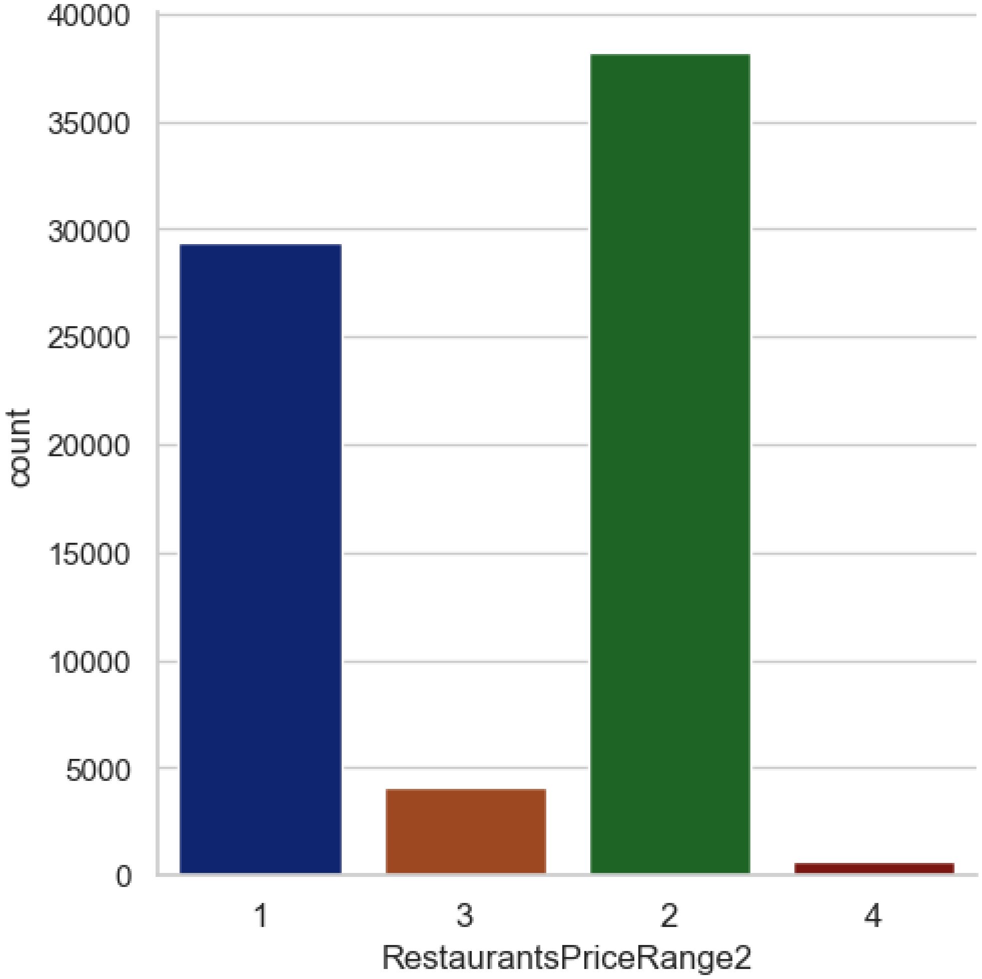
	RestaurantsPriceRange2	business_id
0	1	29430
1	2	38180
2	3	4084
3	4	672

In [29]:

```
sns.catplot(x="RestaurantsPriceRange2",
            data=restaurants, kind="count" )
```

Out[29]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a1e3ac8>
```



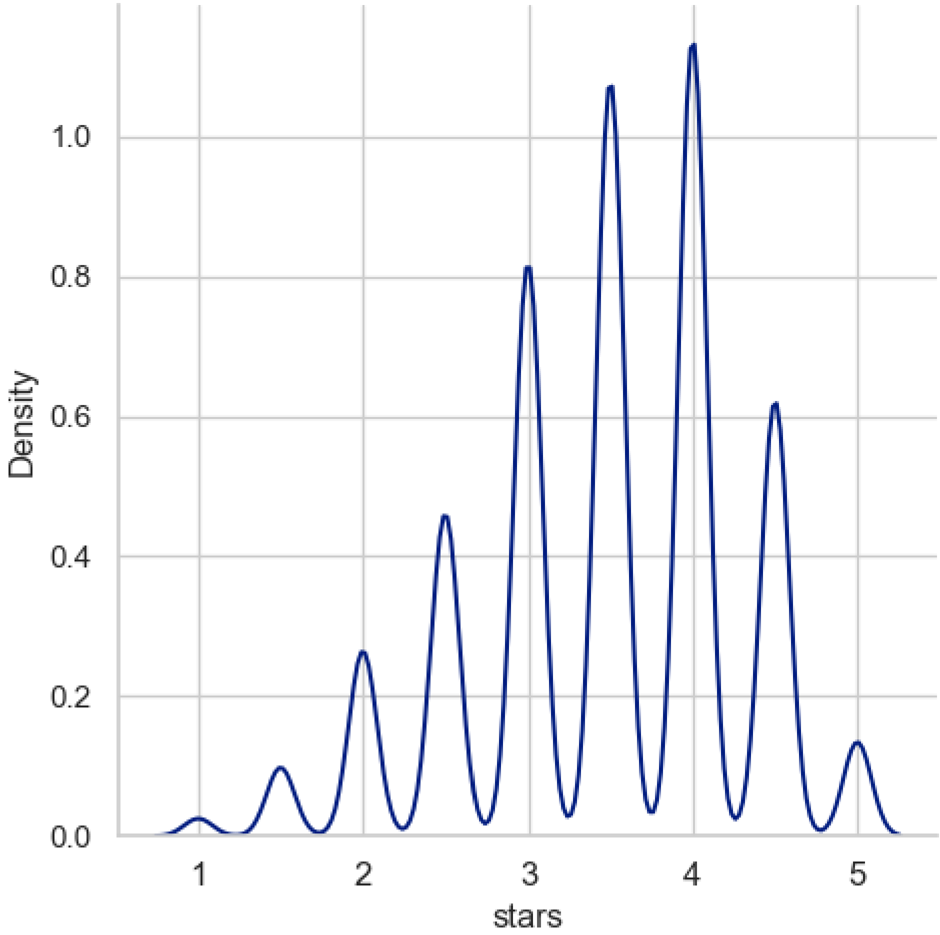
We see that most of the restaurants are in the lower price range (1,2) and lesser in high end (3) and premium (4)

In [30]:

```
sns.displot(restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
            x="stars", kind="kde")
```

Out[30]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a25ce48>
```



Most of the restaurants are between 3 to 4 stars

In [31]:

```
restaurants.loc[restaurants["RestaurantsPriceRange2"].notnull(),"stars"].describe()
```

Out[31]:

```
count    72366.00
mean     3.47
std      0.81
min      1.00
25%      3.00
50%      3.50
75%      4.00
max      5.00
Name: stars, dtype: float64
```

Lets look at stars vs price

In [32]:

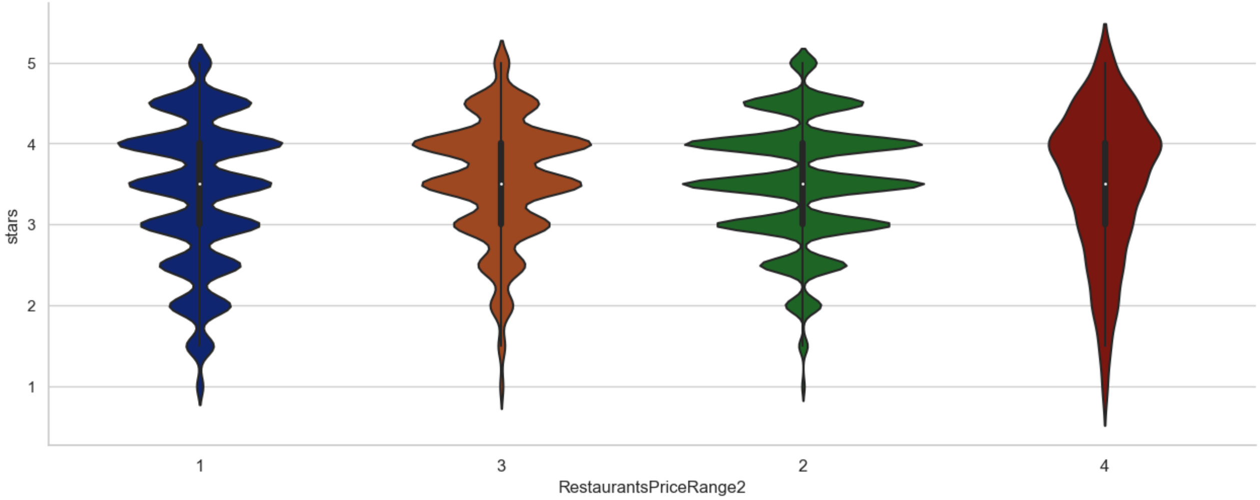
```
# sns.catplot(x="RestaurantsPriceRange2", hue="stars",
#                   data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
#                                  kind="count")

sns.factorplot(x='RestaurantsPriceRange2',
               y='stars' ,
               data=restaurants[(restaurants["RestaurantsPriceRange2"].notnull()) ] ,
               kind='violin', aspect=2.5)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3714: UserWarning: Th
e `factorplot` function has been renamed to `catplot`. The original name will be removed
in a future release. Please update your code. Note that the default `kind` in `factorplo
```

```
t` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

Out[32]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a291208>
```



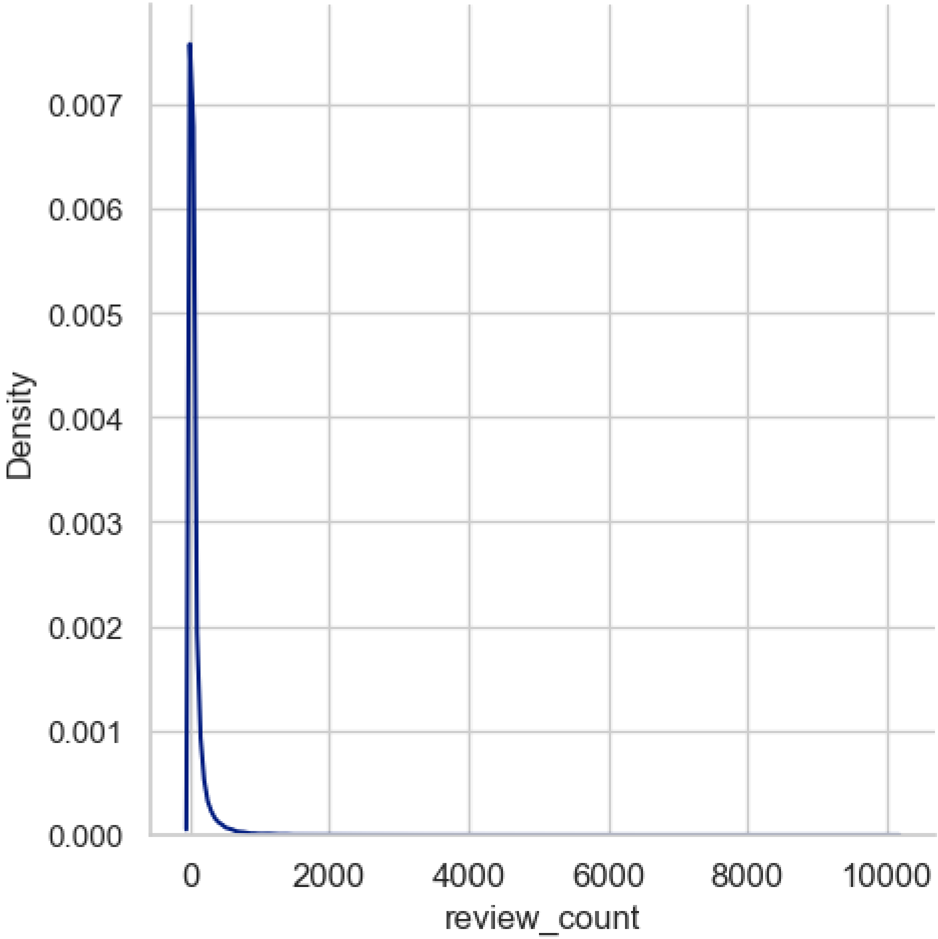
restaurant ratings & price ranges are not necesarily consistent as the stars count is high towards 4 stars with an exception for premium restaurants (4) where their ratings skew higher

In [33]:

```
sns.displot(restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
            x="review_count", kind="kde")
```

Out[33]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a2cb3c8>
```



Looks like review count has a long tail. Lets get basic decriptive stats for this feature

In [34]:

```
restaurants.loc[restaurants["RestaurantsPriceRange2"].notnull(),"review_count"].describ
```

Out[34]:

```
count    72366.00
mean        73.79
std        190.75
min          3.00
25%          9.00
50%         23.00
75%         67.00
max      10129.00
Name: review_count, dtype: float64
```

In [35]:

```
restaurants.loc[restaurants["RestaurantsPriceRange2"].notnull(),"review_count"].quantil
```

Out[35]:

```
460.875
```

To understand the review count distribution we'll cap the review count to 97.5 precentile

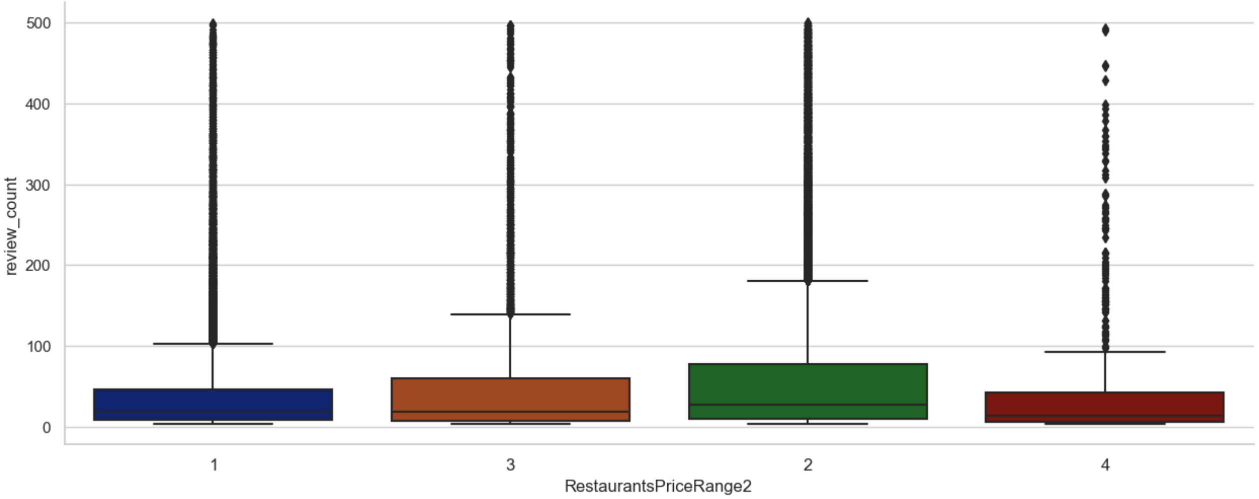
In [36]:

```
sns.factorplot(x='RestaurantsPriceRange2',
               y='review_count' ,
               data=restaurants[(restaurants["RestaurantsPriceRange2"].notnull()) &
                                (restaurants["review_count"]<=500)] , kind='box', aspect
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3714: UserWarning: Th
e `factorplot` function has been renamed to `catplot`. The original name will be removed
in a future release. Please update your code. Note that the default `kind` in `factorplo
t` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

Out[36]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a3307c8>
```



Though the premium restaurants (4) are reviewed lesser as compared to other categories of restaurants, l ooks like there are quite a few exceptional places who have got good amount of reviews

In [37]:

```
restaurants["NoiseLevel"]=None
```

In [38]:

```
restaurants.loc[restaurants["attribute_exists"]=="dict","NoiseLevel"] = [d.get('NoiseLe
```

In [39]:

```
restaurants["NoiseLevel"]=restaurants["NoiseLevel"].str.replace("u\'","\'")
```

In [40]:

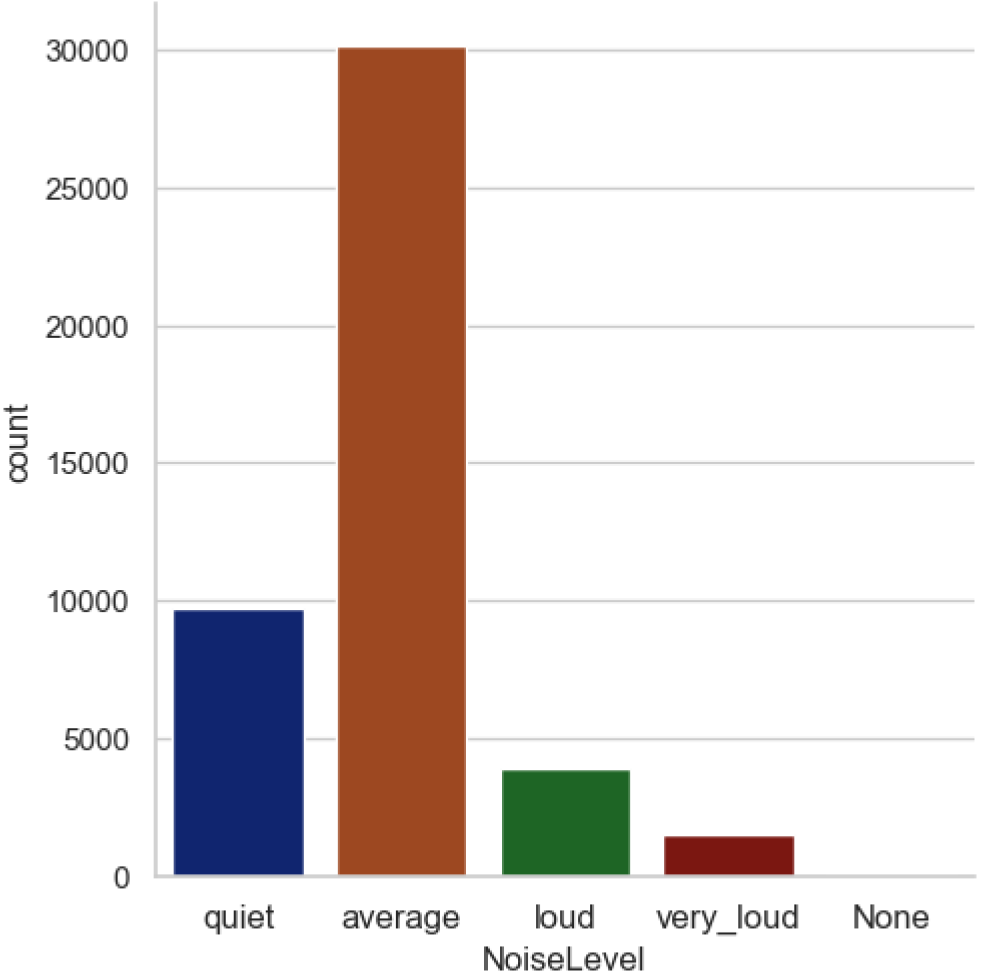
```
restaurants["NoiseLevel"]=restaurants["NoiseLevel"].str.replace("\'","")
```

In [41]:

```
sns.catplot(x="NoiseLevel",
            data=restaurants, kind="count" )
```

Out[41]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a9cdac8>
```



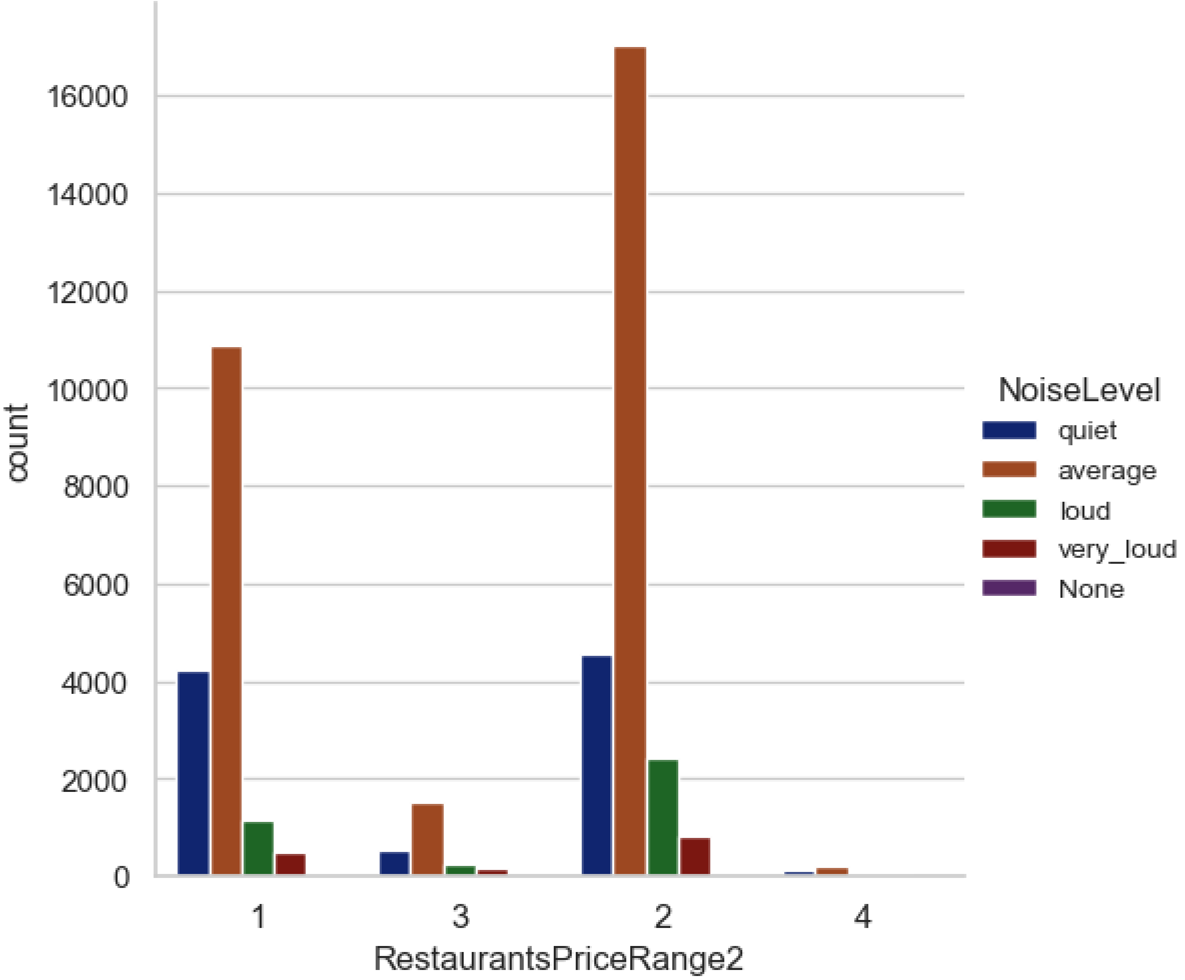
Most of the restaurants seems to have average noise level

In [42]:

```
sns.catplot(x="RestaurantsPriceRange2",  hue="NoiseLevel",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[42]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a97fe88>
```



In [43]:

```
restaurants["CoatCheck"]=None
```

In [44]:

```
restaurants.loc[restaurants["attribute_exists"]=="dict","CoatCheck"] = [d.get('CoatChec
```

In [45]:

```
restaurants.loc[restaurants["CoatCheck"]=="None","CoatCheck"]=None
```

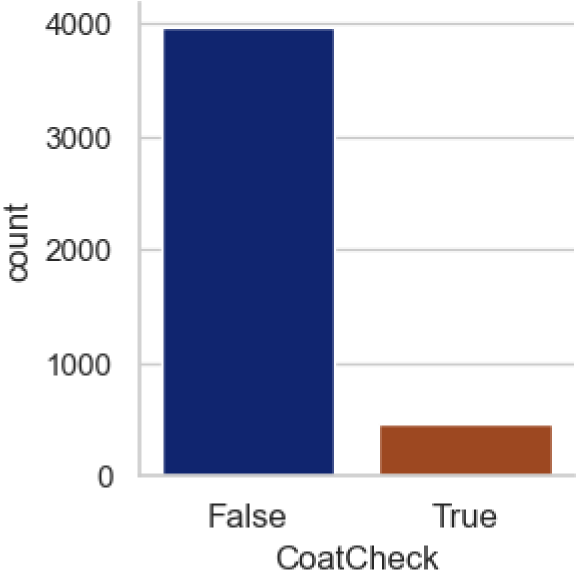
In [46]:

```
sns.catplot(x="CoatCheck",
            data=restaurants, kind="count", size=3 )
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[46]:

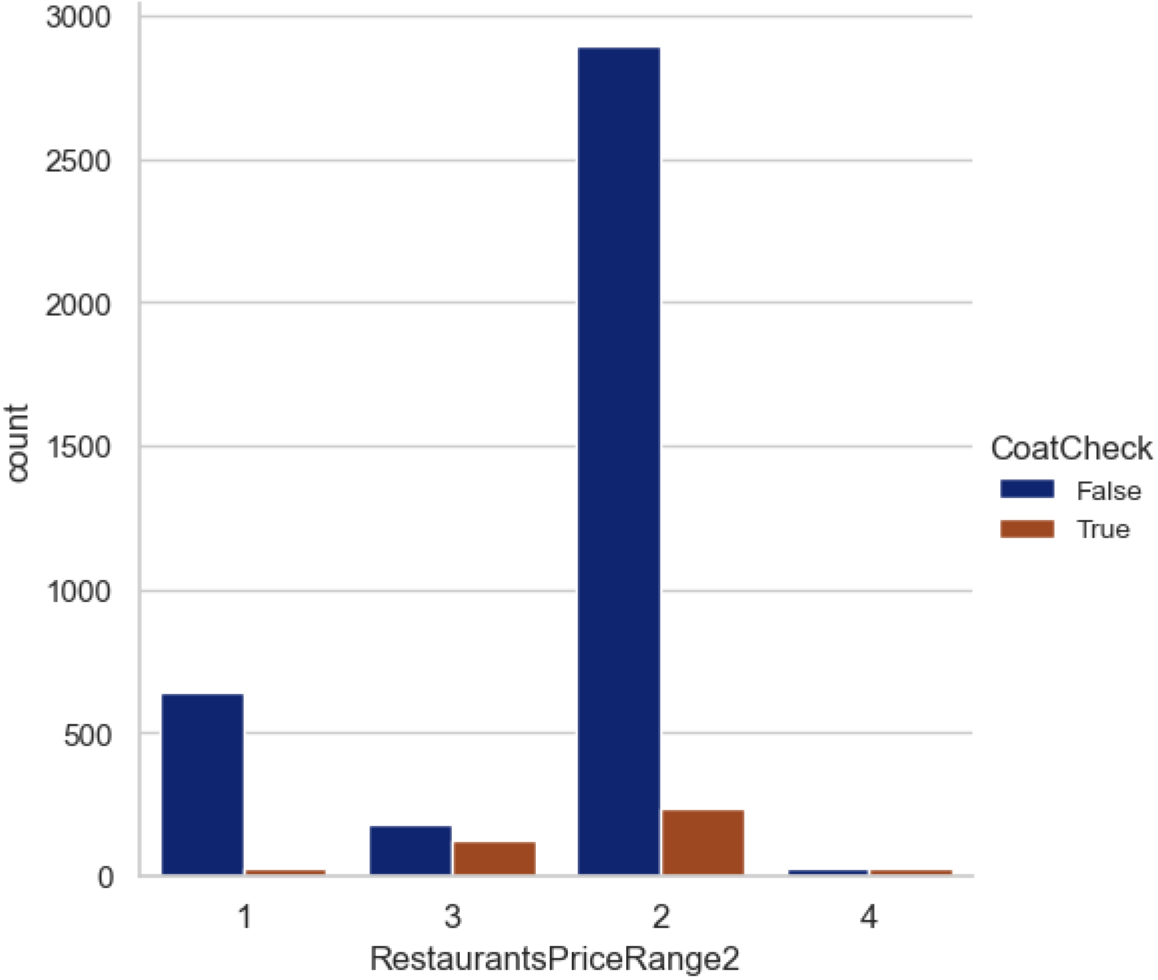
```
<seaborn.axisgrid.FacetGrid at 0x2422a94ac48>
```



Most of the restaurants don't seem to have coatcheck

In [47]:
```
sns.catplot(x="RestaurantsPriceRange2",  hue="CoatCheck",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[47]:
```
<seaborn.axisgrid.FacetGrid at 0x2422a8c8608>
```



In [48]:

```
restaurants["Alcohol"]=None
```

In [49]:

```
restaurants.loc[restaurants["attribute_exists"]=="dict","Alcohol"] = [d.get('Alcohol')
```

In [50]:

```
restaurants["Alcohol"]=restaurants["Alcohol"].str.replace("u\'","\'")
```

In [51]:

```
restaurants.loc[restaurants["Alcohol"]=="\'none\'","Alcohol"]="NoAlcohol"
```

In [52]:

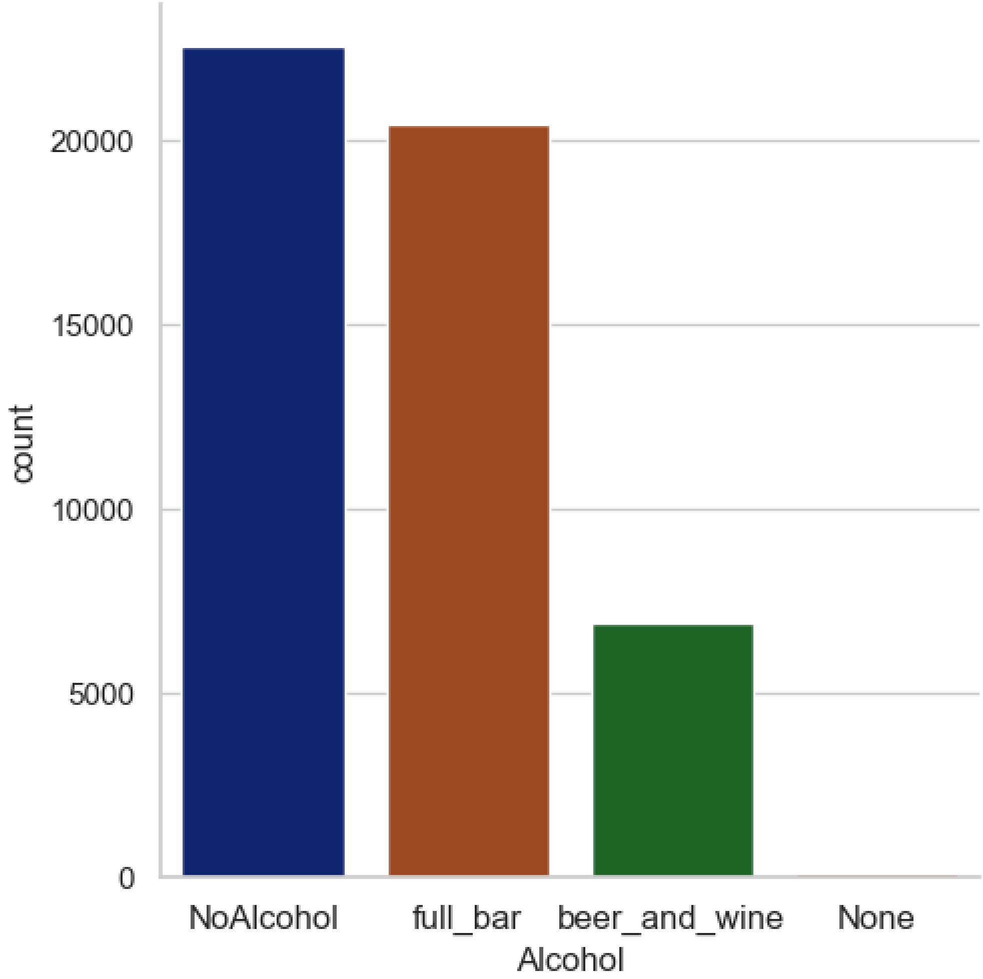
```
restaurants["Alcohol"]=restaurants["Alcohol"].str.replace("\'","")
```

In [53]:

```
sns.catplot(x="Alcohol",
            data=restaurants, kind="count")
```

Out[53]:

```
<seaborn.axisgrid.FacetGrid at 0x2422b2ef448>
```

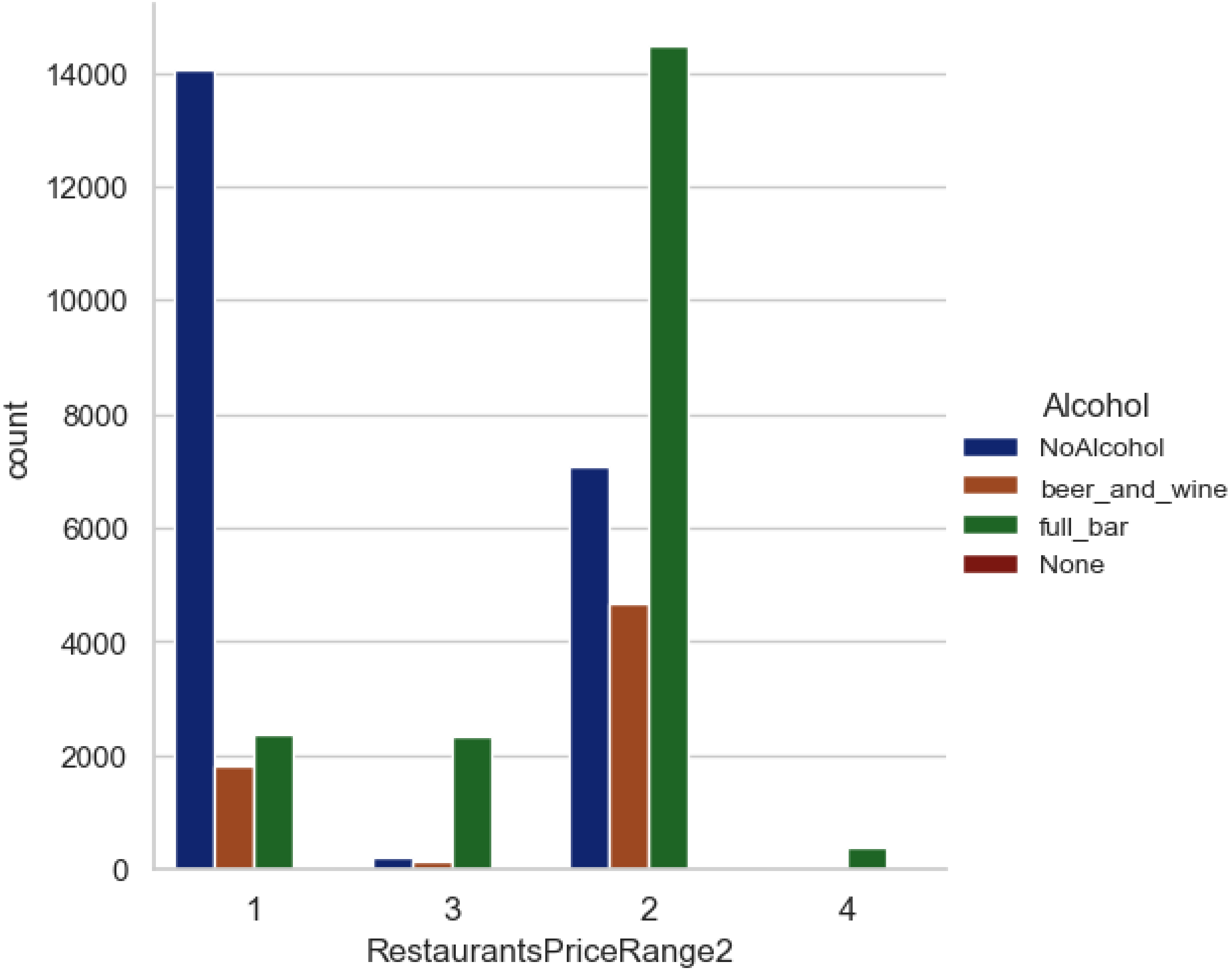


Most of the restaurants don't serve alcohol

In [54]:

```
sns.catplot(x="RestaurantsPriceRange2",  hue="Alcohol",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[54]:
```
<seaborn.axisgrid.FacetGrid at 0x2422b27ea48>
```



High End (3) & Premium(4) restaurants mostly have a full bar

In [55]:
```
restaurants["BusinessAcceptsCreditCards"]=None
```

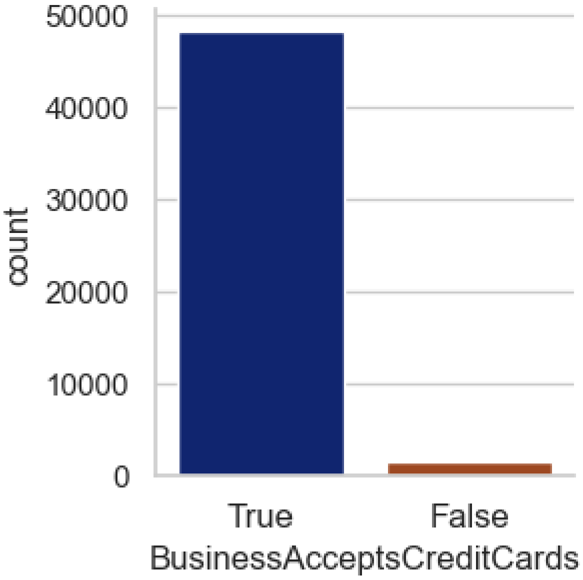
In [56]:
```
restaurants.loc[restaurants["attribute_exists"]=="dict","BusinessAcceptsCreditCards"] =
```

In [57]:
```
restaurants.loc[restaurants["BusinessAcceptsCreditCards"]=="None","BusinessAcceptsCredi
```

In [58]:
```
sns.catplot(x="BusinessAcceptsCreditCards",
            data=restaurants, kind="count", size=3)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[58]:
```
<seaborn.axisgrid.FacetGrid at 0x2422b1e1f88>
```



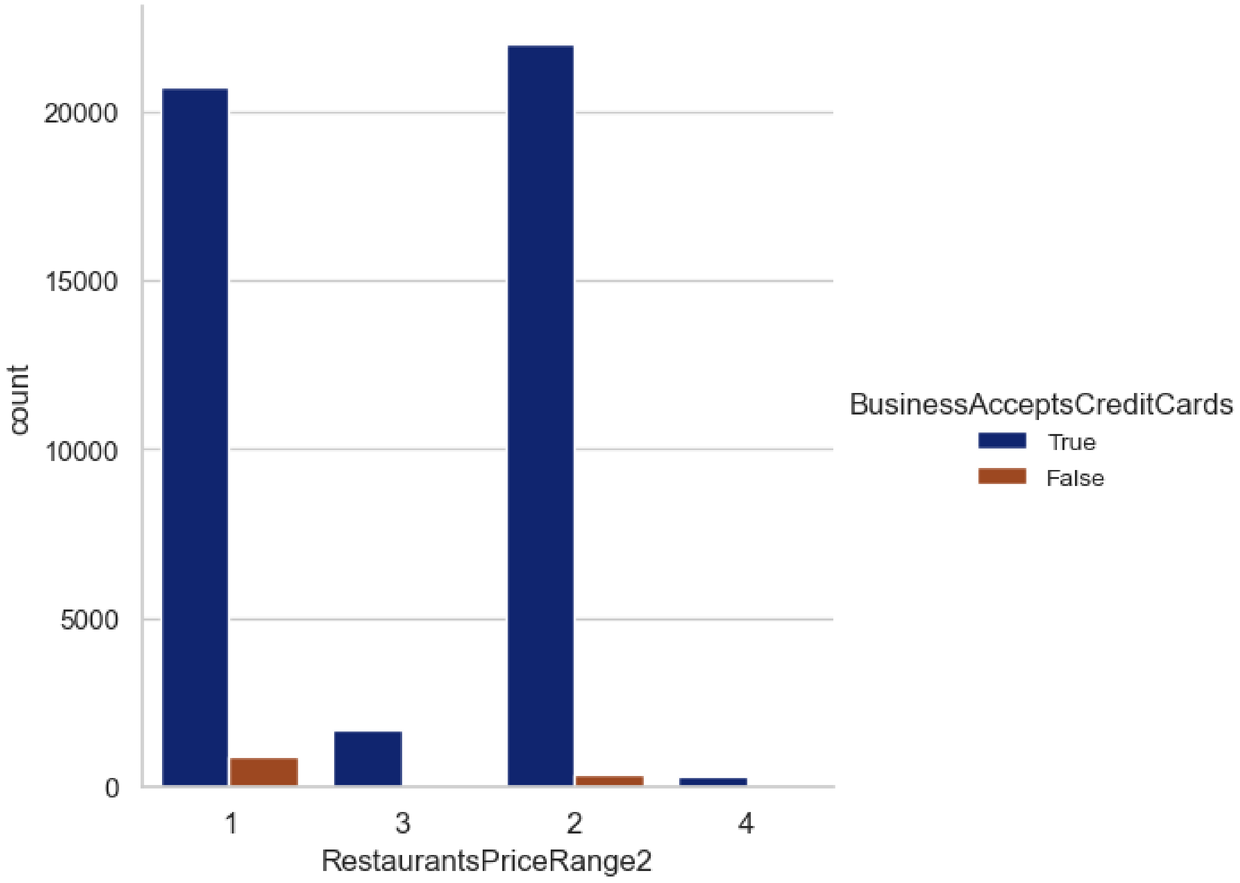
Most of the restaurants accept credit cards

In [59]:

```
sns.catplot(x="RestaurantsPriceRange2",  hue="BusinessAcceptsCreditCards",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[59]:

```
<seaborn.axisgrid.FacetGrid at 0x2422b2d3408>
```



all the high end & premium restaurants accept credit cards

In [60]:

```
restaurants["GoodForKids"]=None
```

In [61]:

```
restaurants.loc[restaurants["attribute_exists"]=="dict","GoodForKids"] = [d.get('GoodFo
```

In [62]:

```
restaurants.loc[restaurants["GoodForKids"]=="None","GoodForKids"]=None
```

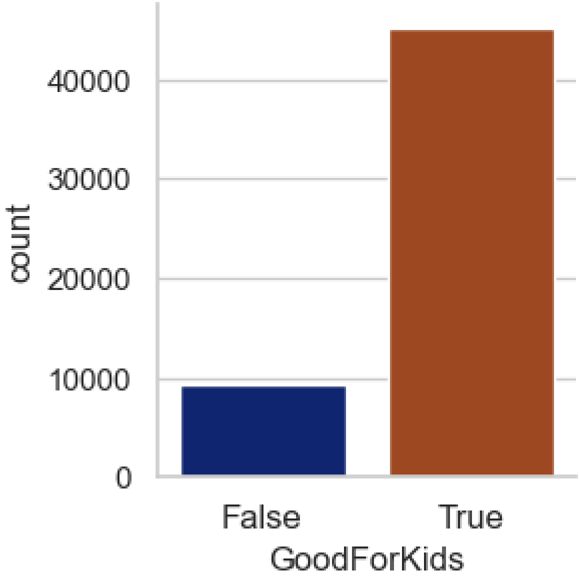
In [63]:

```
sns.catplot(x="GoodForKids",
            data=restaurants, kind="count", size=3)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[63]:

```
<seaborn.axisgrid.FacetGrid at 0x2422b17ca48>
```



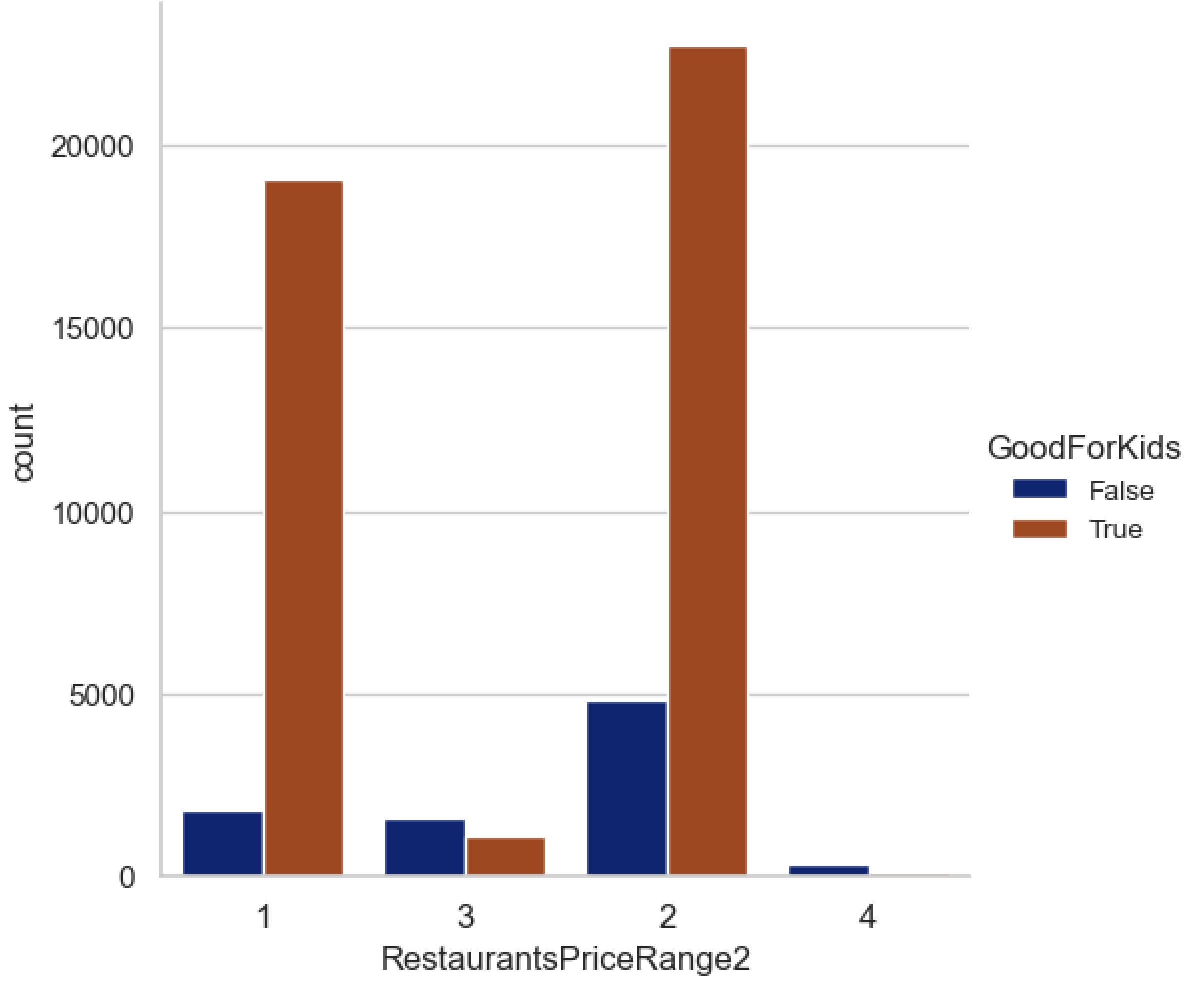
Most of the restaurants are Kids Friendly

In [64]:

```
sns.catplot(x="RestaurantsPriceRange2",  hue="GoodForKids",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[64]:

```
<seaborn.axisgrid.FacetGrid at 0x2422a3f6408>
```



Premium restaurants are not kids freindly (maybe because of the full bar)

```
In [65]: restaurants["RestaurantsDelivery"]=None
```

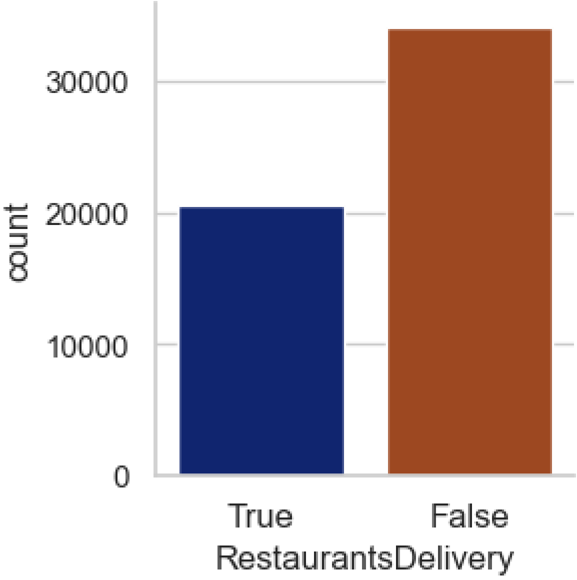
```
In [66]: restaurants.loc[restaurants["attribute_exists"]=="dict","RestaurantsDelivery"] = [d.get
```

```
In [67]: restaurants.loc[restaurants["RestaurantsDelivery"]=="None","RestaurantsDelivery"]=None
```

```
In [68]: sns.catplot(x="RestaurantsDelivery",
                     data=restaurants, kind="count", size=3)
```

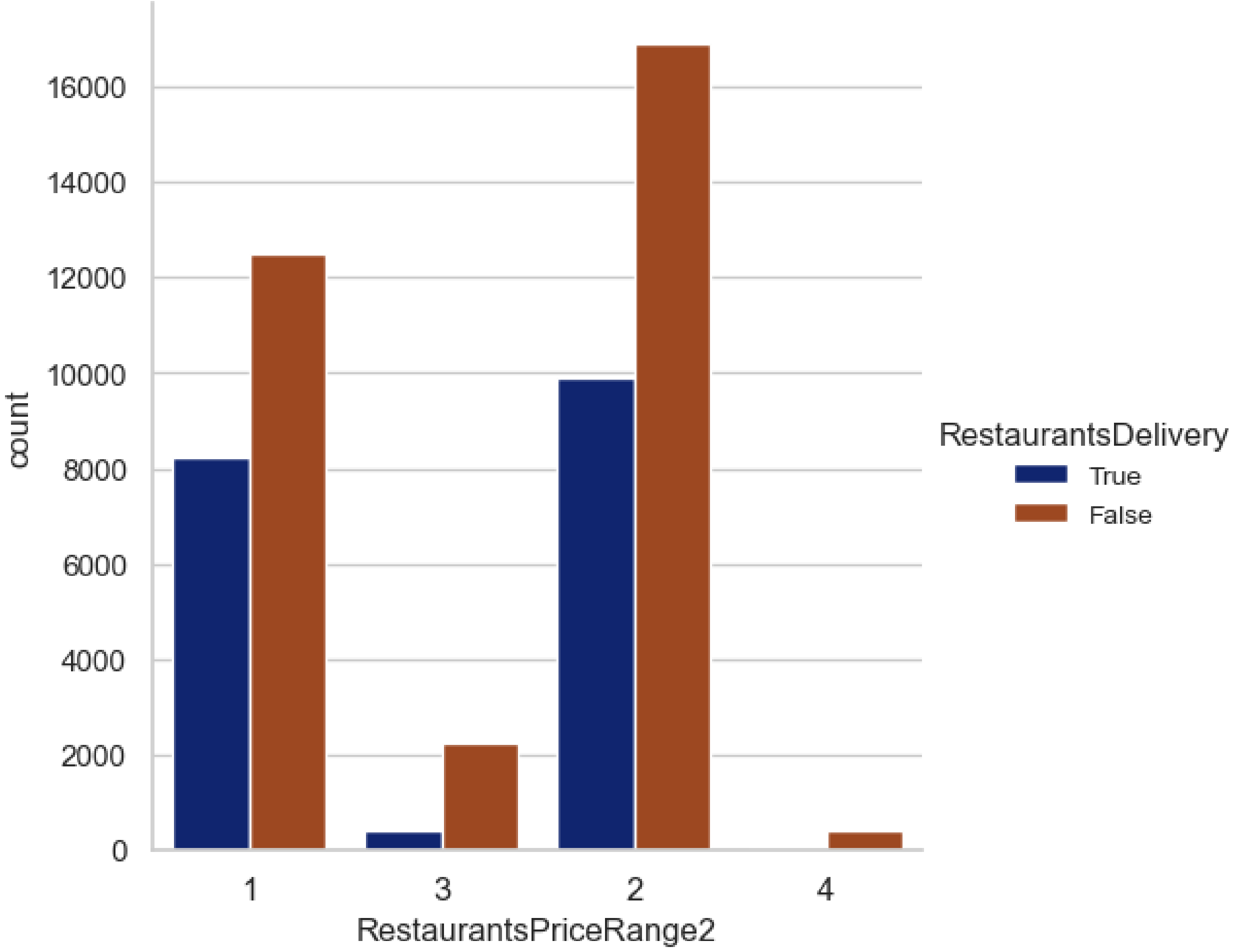
```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

```
Out[68]: <seaborn.axisgrid.FacetGrid at 0x2422a314e88>
```



In [69]:
```
sns.catplot(x="RestaurantsPriceRange2",  hue="RestaurantsDelivery",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[69]:
```
<seaborn.axisgrid.FacetGrid at 0x2422a4ef1c8>
```



High end & Premium restaurants mostly don't offer delivery

In [70]:
```
restaurants["RestaurantsTakeOut"]=None
```

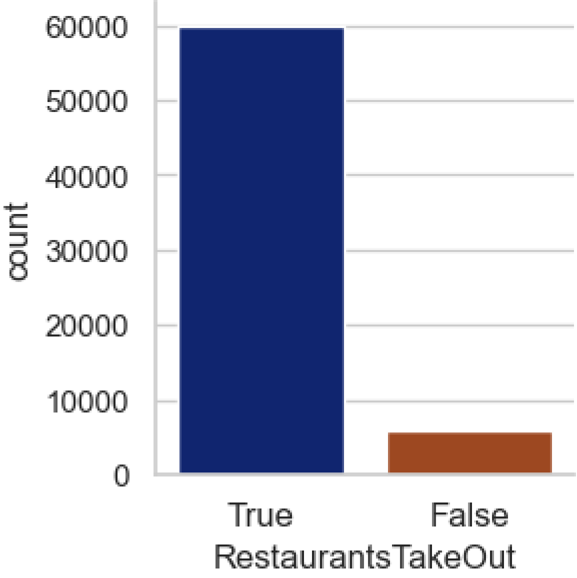
In [71]:
```
restaurants.loc[restaurants["attribute_exists"]=="dict","RestaurantsTakeOut"] = [d.get(
```

In [72]:
```
restaurants.loc[restaurants["RestaurantsTakeOut"]=="None","RestaurantsTakeOut"]=None
```

In [73]:
```
sns.catplot(x="RestaurantsTakeOut",
            data=restaurants, kind="count", size=3)
```

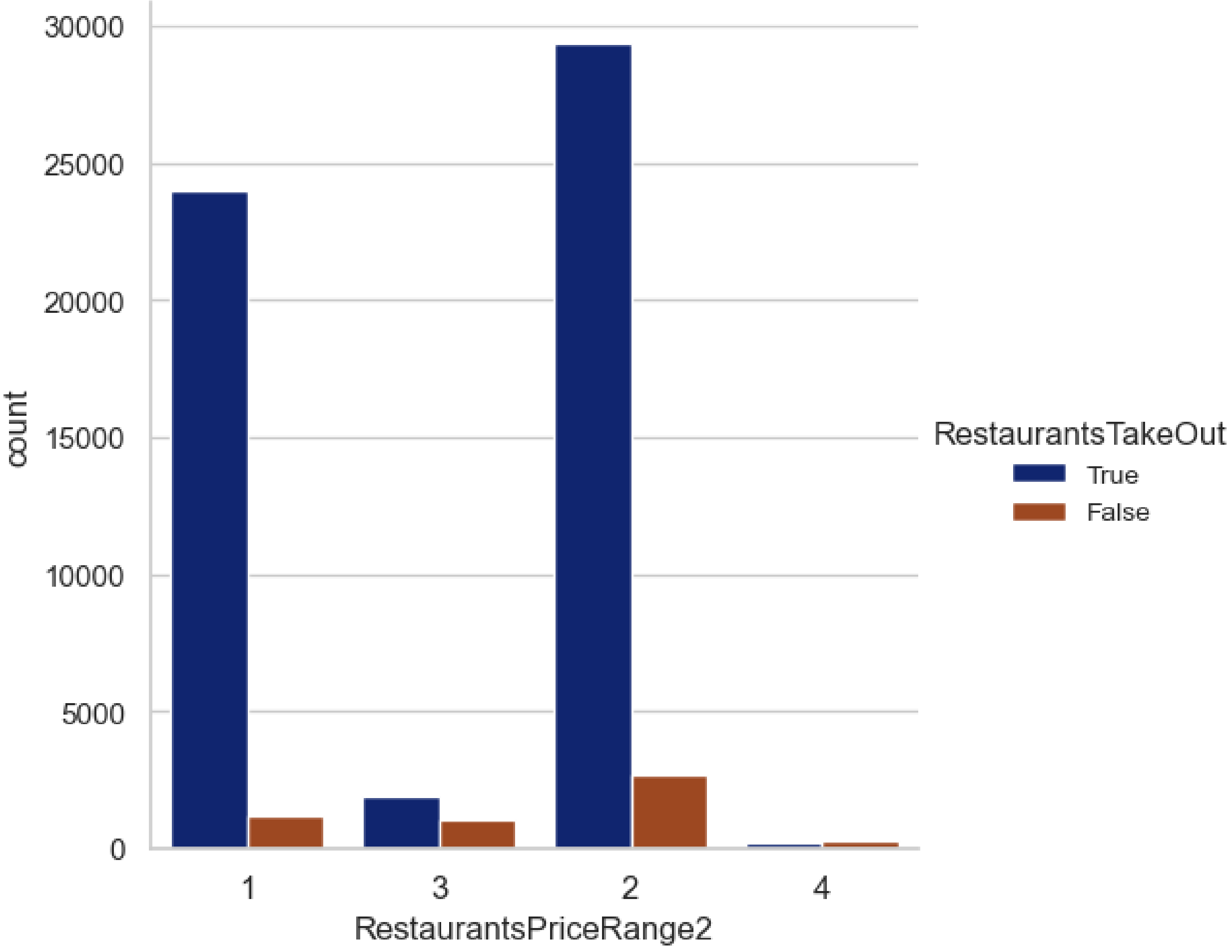
```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[73]:
```
<seaborn.axisgrid.FacetGrid at 0x2422a4ddc48>
```



In [74]:
```
sns.catplot(x="RestaurantsPriceRange2",  hue="RestaurantsTakeOut",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[74]:
```
<seaborn.axisgrid.FacetGrid at 0x2422a5ef748>
```



Though high end & premium don't offer much of a delivery, they do offer takeout"

In [75]:
```
restaurants["OutdoorSeating"]=None
```

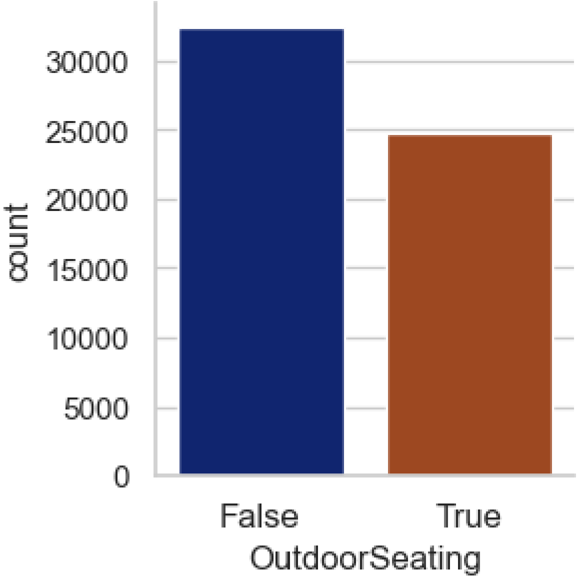
In [76]:
```
restaurants.loc[restaurants["attribute_exists"]=="dict","OutdoorSeating"] = [d.get('Out
```

In [77]:
```
restaurants.loc[restaurants["OutdoorSeating"]=="None","OutdoorSeating"]=None
```

In [78]:
```
sns.catplot(x="OutdoorSeating",
            data=restaurants, kind="count", size=3)
```

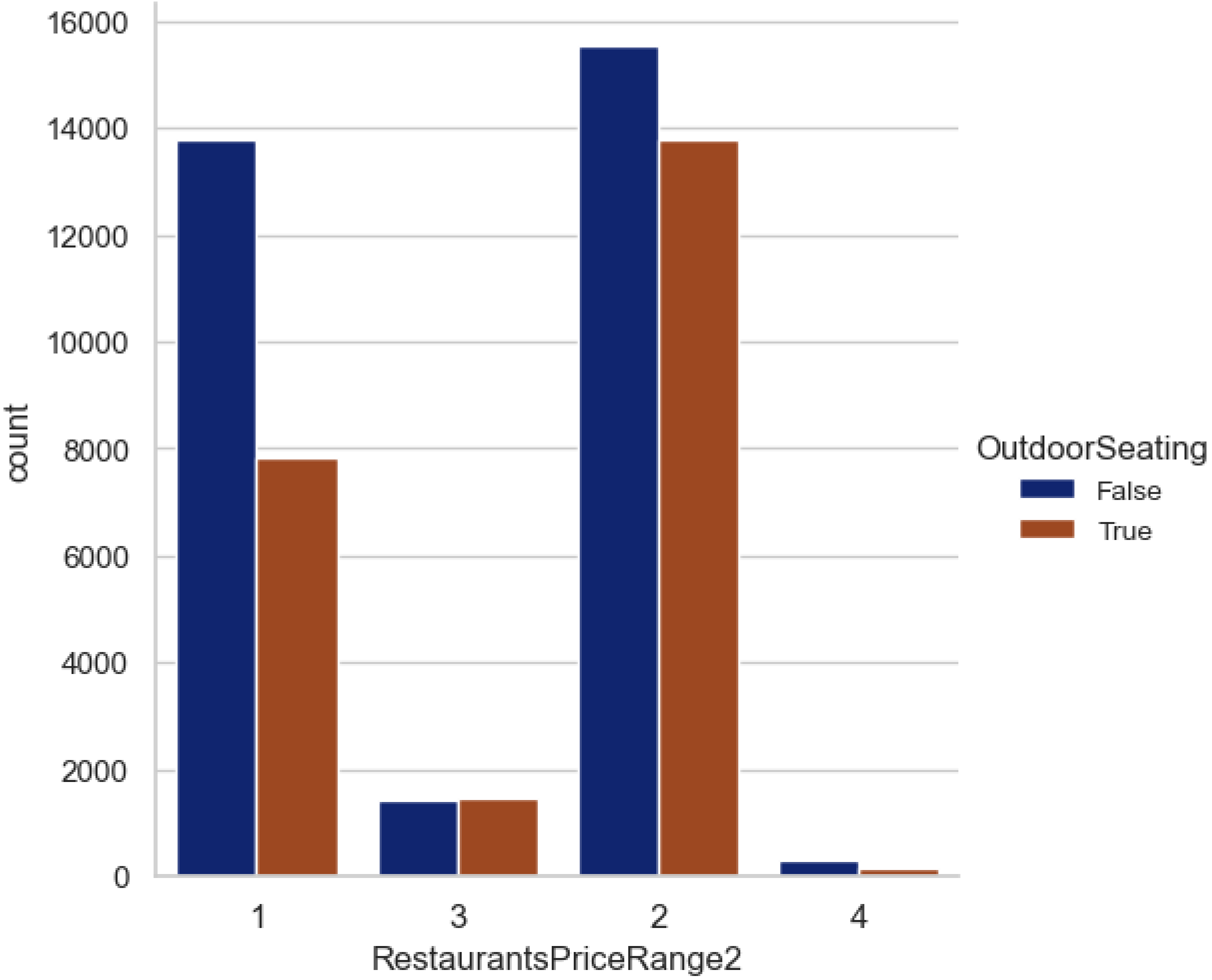
```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[78]:
```
<seaborn.axisgrid.FacetGrid at 0x2422a6702c8>
```



In [79]:
```
sns.catplot(x="RestaurantsPriceRange2",  hue="OutdoorSeating",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[79]:
```
<seaborn.axisgrid.FacetGrid at 0x2422a6e1508>
```



No concrete inference with this feature

In [80]:
```
restaurants["WiFi"]=None
```

In [81]:
```
restaurants.loc[restaurants["attribute_exists"]=="dict","WiFi"] = [d.get('WiFi')  for d
```

In [82]:
```
restaurants.loc[restaurants["WiFi"]=="None","WiFi"]=None
```

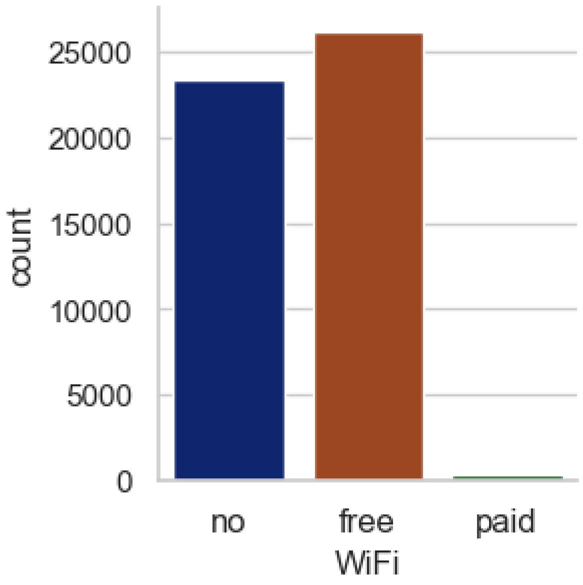
In [83]:
```
restaurants["WiFi"]=restaurants["WiFi"].str.replace("u\'","\'")
```

In [84]:
```
restaurants["WiFi"]=restaurants["WiFi"].str.replace("\'","")
```

In [85]:
```
sns.catplot(x="WiFi",
            data=restaurants, kind="count", size=3)
```

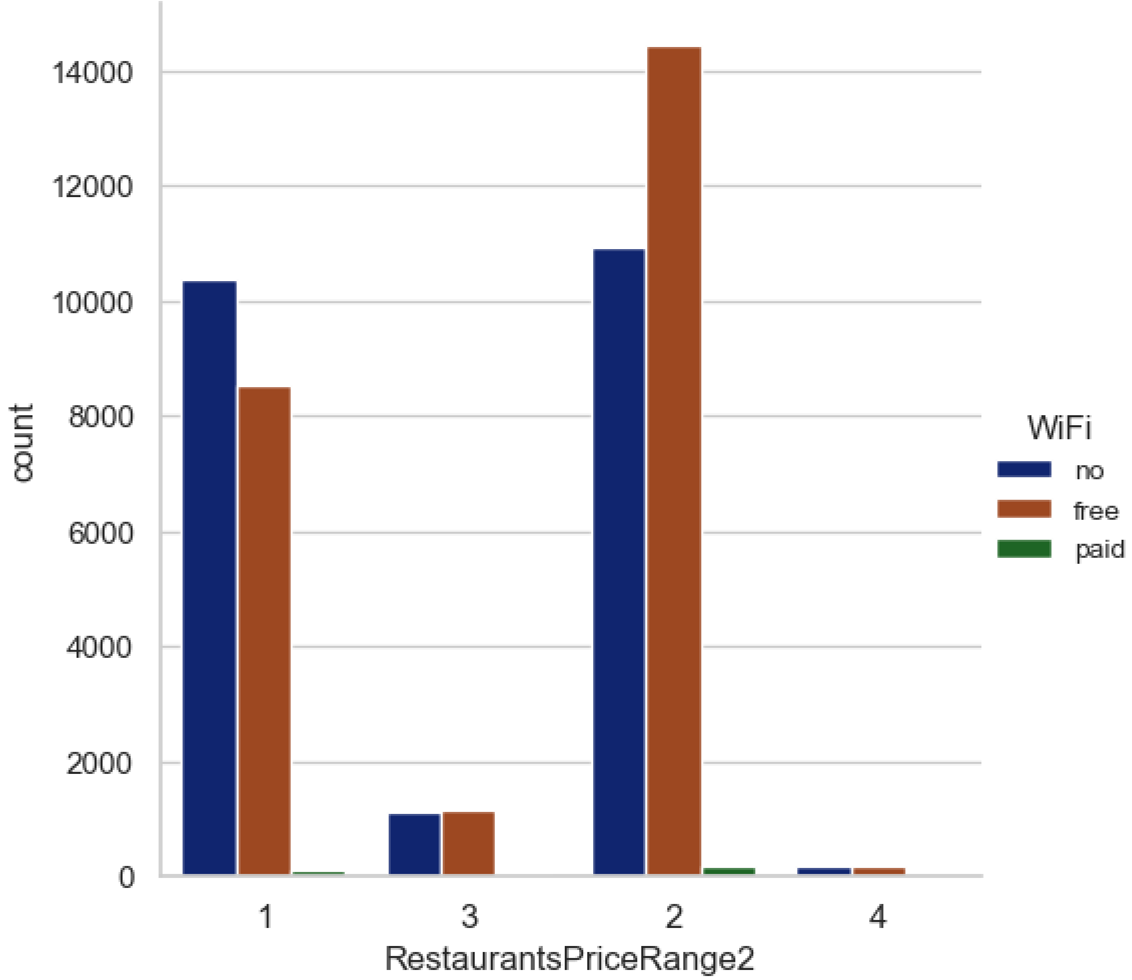
```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[85]:
```
<seaborn.axisgrid.FacetGrid at 0x2422b99e5c8>
```



In [86]:
```
sns.catplot(x="RestaurantsPriceRange2",  hue="WiFi",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[86]:
```
<seaborn.axisgrid.FacetGrid at 0x2422b95cb88>
```



No concrete inference for this feature

if restaurant is a chain

In [87]:

```
rest_type_chain=restaurants.groupby(["name"])["business_id"].count().reset_index().sort
```

In [88]:

```
rest_type_chain["Is_chain"]=np.where(rest_type_chain["business_id"]>1, True,False)
```

In [89]:

```
restaurants=pd.merge(restaurants,rest_type_chain[["name","Is_chain"]], on="name", how="
```

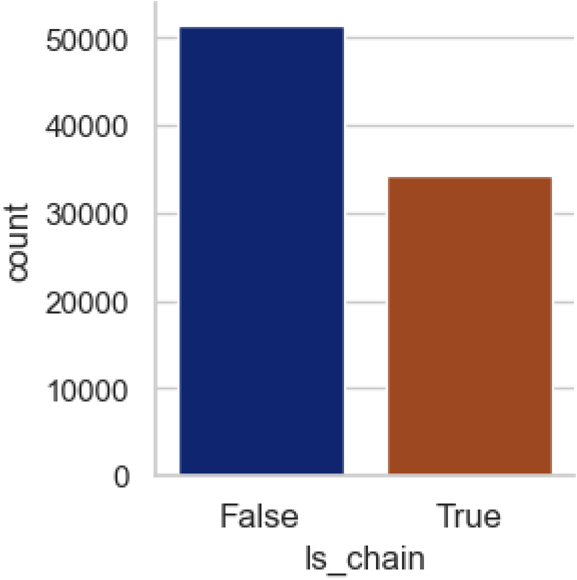
In [90]:

```
sns.catplot(x="Is_chain",
            data=restaurants, kind="count", size=3)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3747: UserWarning: Th
e `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[90]:

```
<seaborn.axisgrid.FacetGrid at 0x2422b91e1c8>
```

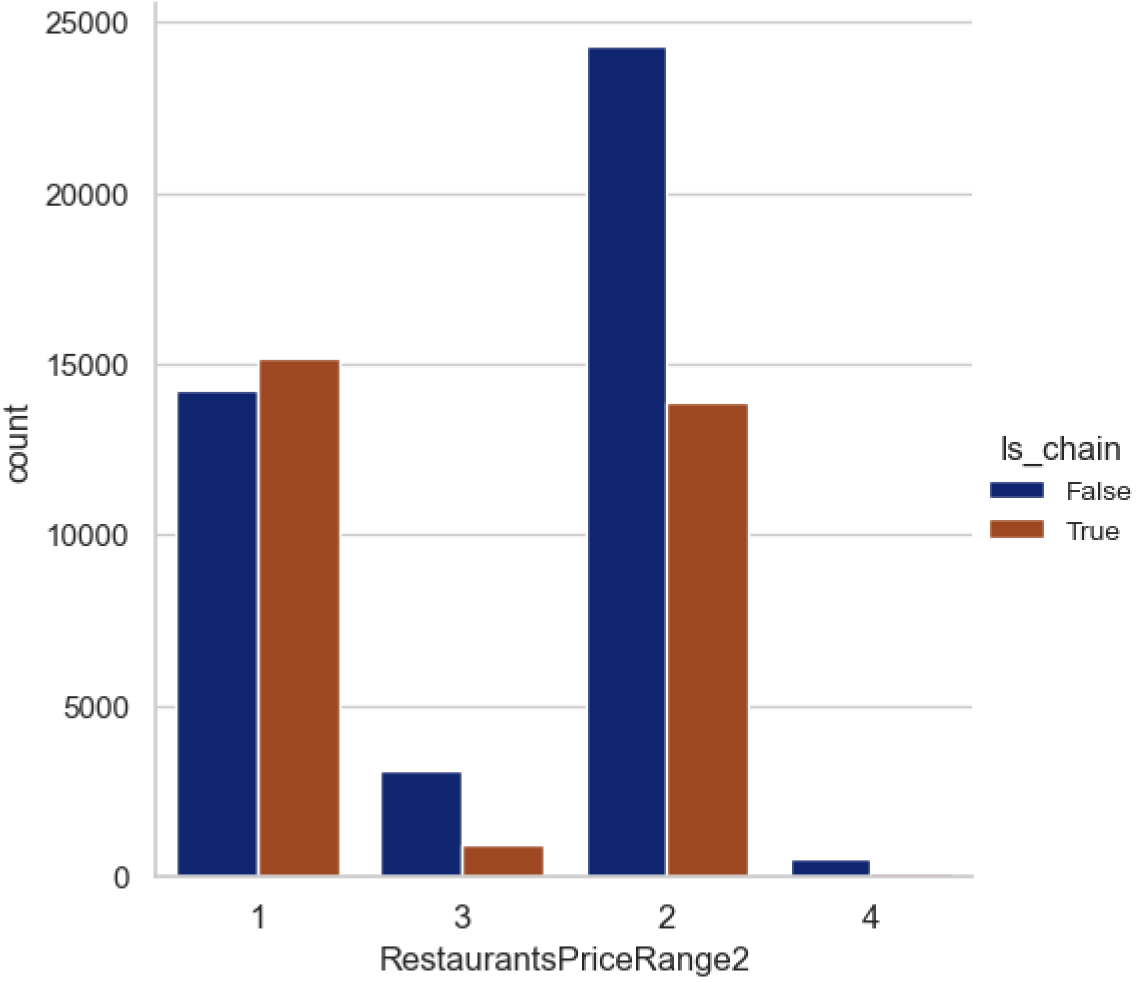


In [91]:

```
sns.catplot(x="RestaurantsPriceRange2",  hue="Is_chain",
                data=restaurants[restaurants["RestaurantsPriceRange2"].notnull()],
                                 kind="count")
```

Out[91]:

```
<seaborn.axisgrid.FacetGrid at 0x2422b8c4dc8>
```



Premium restaurants are standalone

In [92]:

```
restaurants.columns
```

Out[92]:

```
Index(['business_id', 'name', 'address', 'city', 'state', 'postal_code',
       'latitude', 'longitude', 'stars', 'review_count', 'is_open',
       'attributes', 'categories', 'hours', 'Is_Restaurant',
       'attribute_exists', 'RestaurantsPriceRange2', 'NoiseLevel', 'CoatCheck',
       'Alcohol', 'BusinessAcceptsCreditCards', 'GoodForKids',
       'RestaurantsDelivery', 'RestaurantsTakeOut', 'OutdoorSeating', 'WiFi',
       'Is_chain'],
      dtype='object')
```

In [93]:

```
#restaurants = pd.read_csv("./data/business_Filtered.csv")
```

Top 10 cities by restaurant review count

In [94]:

```
restaurants.groupby('city')['review_count'].sum().reset_index().rename(columns={'review
```

Out[94]:

	city	count
341	Las Vegas	1634289
589	Phoenix	547112
803	Toronto	485445
731	Scottsdale	291317
117	Charlotte	280520
600	Pittsburgh	210514
464	Montréal	152937
795	Tempe	152896
280	Henderson	152587
429	Mesa	118260

In [95]:

```
restaurants[restaurants["RestaurantsPriceRange2"].notnull()].to_csv("./data/business_Fi
```

In [96]:

```
relevent_restaurants = restaurants[restaurants["RestaurantsPriceRange2"].notnull()]
```

In [97]:

```
relevent_restaurants.shape
```

Out[97]:

```
(72366, 27)
```

For Recommendation Engine purpose I'll select only top 3 cities due to hardware limitations in my local machine

In [98]:

```
#freeing up memory
#del(business1, business)
```

```
del(restaurants)
gc.collect()
```

Out[98]:
```
20
```

Read & Exploring Reviews data

In [99]:
```
%%time
review1 = []
with open('./data/yelp_academic_dataset_review.json', 'r', encoding='utf-8') as f:
#with open('./data/review_filtered.json','r',encoding='utf-8') as f:
    for line in f:
        review1.append(json.loads(line))
print(review1[0])
```

```
{'review_id': 'xQY8N_XvtGbearJ5X4QryQ', 'user_id': 'OwjRMXRC0KyPrIlcjaXeFQ', 'business_i
d': '-MhfebM0QIsKt87iDN-FNw', 'stars': 2.0, 'useful': 5, 'funny': 0, 'cool': 0, 'text':
'As someone who has worked with many museums, I was eager to visit this gallery on my mo
st recent trip to Las Vegas. When I saw they would be showing infamous eggs of the House
of Faberge from the Virginia Museum of Fine Arts (VMFA), I knew I had to go!\n\nTucked a
way near the gelateria and the garden, the Gallery is pretty much hidden from view. It
\'s what real estate agents would call "cozy" or "charming" - basically any euphemism fo
r small.\n\nThat being said, you can still see wonderful art at a gallery of any size, s
o why the two *s you ask? Let me tell you:\n\n* pricing for this, while relatively inexp
ensive for a Las Vegas attraction, is completely over the top. For the space and the amo
unt of art you can fit in there, it is a bit much.\n* it\'s not kid friendly at all. Ser
iously, don\'t bring them.\n* the security is not trained properly for the show. When th
e curating and design teams collaborate for exhibitions, there is a definite flow. That
means visitors should view the art in a certain sequence, whether it be by historical pe
riod or cultural significance (this is how audio guides are usually developed). When I a
rrived in the gallery I could not tell where to start, and security was certainly not he
lpful. I was told to "just look around" and "do whatever." \n\nAt such a *fine* institut
ion, I find the lack of knowledge and respect for the art appalling.', 'date': '2015-04-
15 05:21:16'}
Wall time: 1min 19s
```

Convert json data to pandas dataframe & filter out reviews of relevent restaurants

In [100…
```
%%time
df_review = pd.DataFrame.from_dict(review1)#.reset_index()#.rename(columns={ 0 :'Count'
```

```
Wall time: 3min 29s
```

In [101…
```
df_review.shape
```

Out[101…
```
(8021122, 9)
```

In [102…
```
del(review1)
gc.collect()
```

Out[102…
```
60
```

In [103…
```
select_columns_to_merge = ['business_id', 'city', 'is_open','RestaurantsPriceRange2',
                                   'NoiseLevel', 'CoatCheck','Alcohol', 'BusinessAcceptsCreditC
                                   'GoodForKids','RestaurantsDelivery', 'RestaurantsTakeOut',
                                   'OutdoorSeating', 'WiFi','Is_chain']
```

In [104…

```
%%time
df_review=pd.merge(df_review, relevent_restaurants[select_columns_to_merge], on="busine
```

```
Wall time: 1min 20s
```

In [105…

```
df_review = df_review[df_review["RestaurantsPriceRange2"].notnull()]
```

In [106…

```
df_review["date"]=pd.to_datetime(df_review["date"])
```

In [107…

```
df_review["date1"]=df_review["date"].dt.date
```

In [108…

```
df_review["days_from_today"]=pd.to_datetime("now").date() - df_review["date1"]
```

In [109…

```
df_review["days_from_today"] = df_review["days_from_today"].dt.days.astype('int')
```

In [110…

```
df_review.shape
```

Out[110…

```
(5527788, 24)
```

Since even after filtering for relevent restaurants we have 5.5 million reviews, I'll sample for 500K rows (which is approx 10% of the reviews) to do the EDA. This choice is due to hardware limitation on my local machine

In [111…

```
df_review_sample = df_review.sample(n = 500000, random_state = 1034)
```

In [112…

```
%%time
df_review.to_csv("./data/review_filtered.csv",encoding='utf-8')
```

```
Wall time: 1min 59s
```

In [113…

```
del(df_review)
gc.collect()
```

Out[113…

```
37
```

In [114…

```
analyser = SentimentIntensityAnalyzer()
```

In [115…

```
%%time
df_review_sample["vader_comp_score"] = df_review_sample["text"].apply(lambda x: analyse
df_review_sample["txt_blb_comp_score"] = df_review_sample["text"].apply(lambda x: TextB
df_review_sample["super_score"] = df_review_sample["stars"] + (df_review_sample["txt_bl
Wall time: 22min 16s
```

VADER(Valence Aware Dictionary and Sentiment Reasoner) sentiment scores & text blob sentiment score are calculated above. VADER is a setiment scoring algorithm tuned towards social media It can happen sometimes that the rating stars are different from the review text. Hence a 'super_score' is calculated where super_score = stars + (vader_compound_score + text_blob_polarity_score)

Both VADER & text blob scores range from -1 to +1 where -1 is highly -ve and +1 is highly positive sentiment This way super score stars will have range from 0 to 6. The stars range from 1 to 5

In [116…

```
df_review_sample.shape
```

Out[116…

```
(500000, 27)
```

In [117…

```
df_review_sample.columns
```

Out[117…

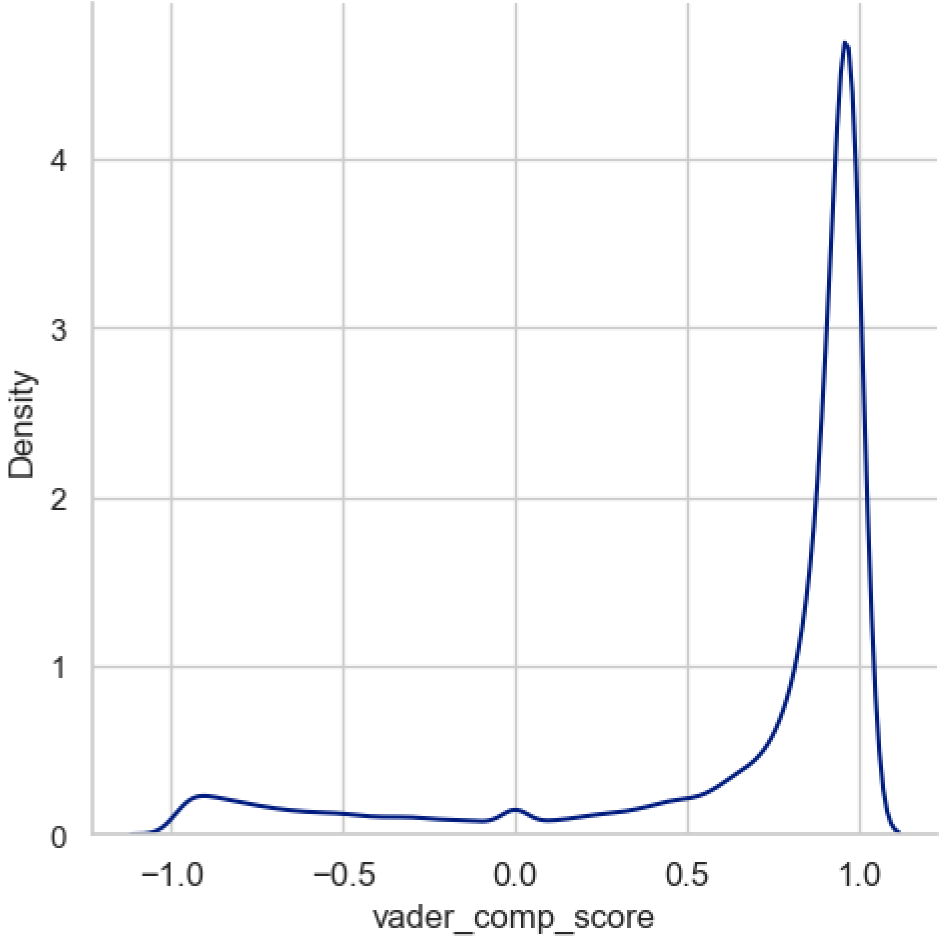
```
Index(['review_id', 'user_id', 'business_id', 'stars', 'useful', 'funny',
       'cool', 'text', 'date', 'city', 'is_open', 'RestaurantsPriceRange2',
       'NoiseLevel', 'CoatCheck', 'Alcohol', 'BusinessAcceptsCreditCards',
       'GoodForKids', 'RestaurantsDelivery', 'RestaurantsTakeOut',
       'OutdoorSeating', 'WiFi', 'Is_chain', 'date1', 'days_from_today',
       'vader_comp_score', 'txt_blb_comp_score', 'super_score'],
      dtype='object')
```

In [118…

```
sns.displot(df_review_sample,
            x="vader_comp_score", kind="kde")
```

Out[118…

```
<seaborn.axisgrid.FacetGrid at 0x2422a54da08>
```



most of the reviews are positive

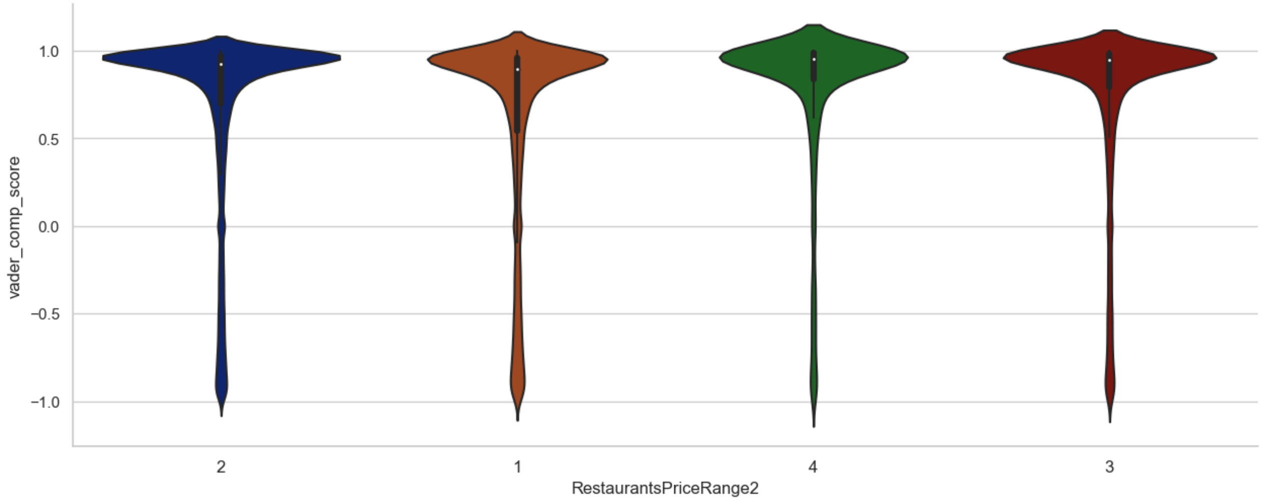
In [119…

```
sns.factorplot(x='RestaurantsPriceRange2',
               y='vader_comp_score' ,
               data = df_review_sample ,
               kind='violin', aspect=2.5)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3714: UserWarning: Th
e `factorplot` function has been renamed to `catplot`. The original name will be removed
in a future release. Please update your code. Note that the default `kind` in `factorplo
t` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

Out[119…

```
<seaborn.axisgrid.FacetGrid at 0x2422e03c048>
```



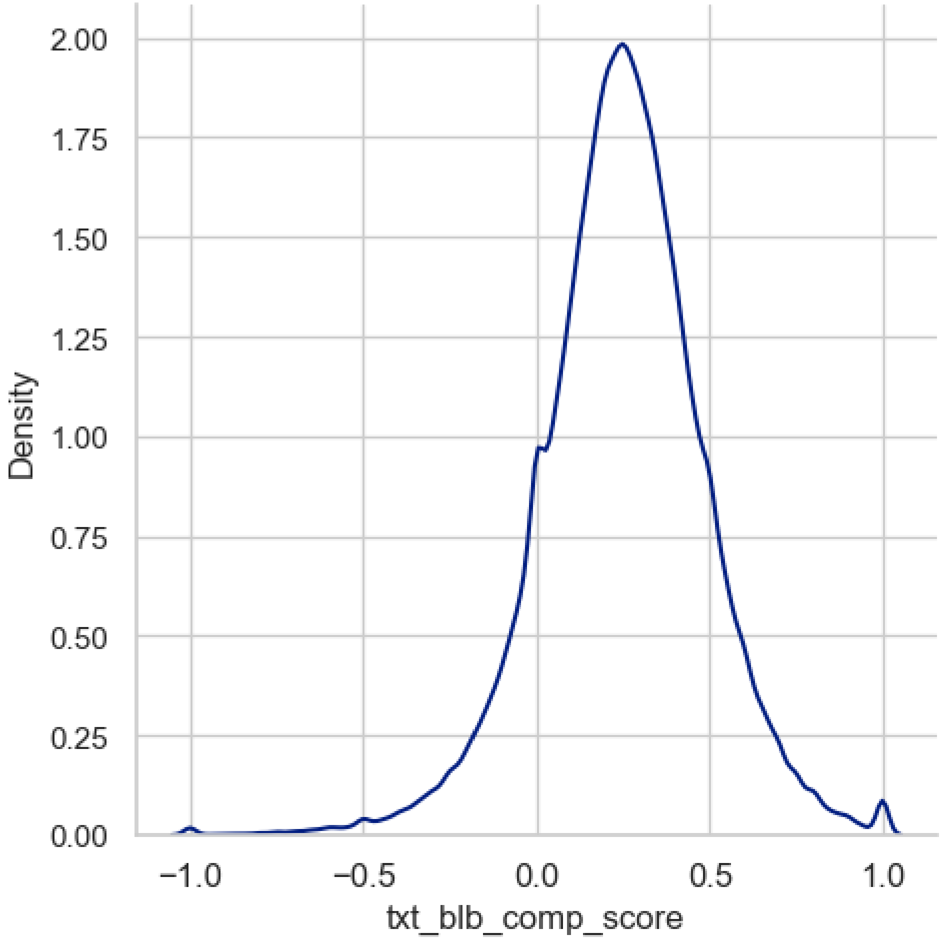
Mostly sentiments scores across all the price ranges are positive. For the lower priced restaurants (1 & 2), it seems they have more -ve sentiments as compared to high end & premium restaurants

In [120…

```
sns.displot(df_review_sample,
            x="txt_blb_comp_score", kind="kde")
```

Out[120…

```
<seaborn.axisgrid.FacetGrid at 0x2422b1a30c8>
```

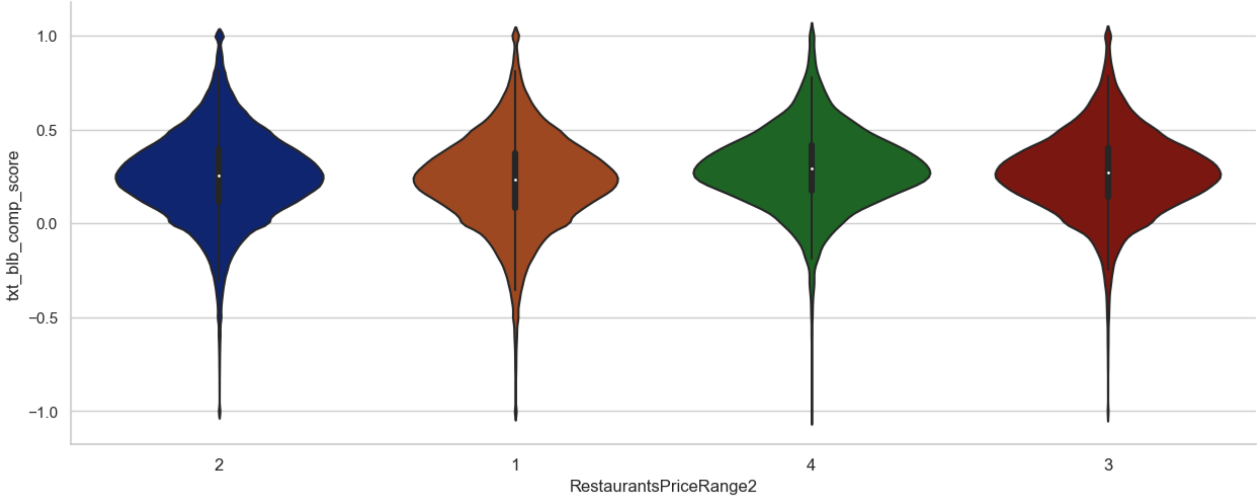


In [121…

```
sns.factorplot(x='RestaurantsPriceRange2',
               y='txt_blb_comp_score' ,
               data = df_review_sample ,
               kind='violin', aspect=2.5)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3714: UserWarning: Th
e `factorplot` function has been renamed to `catplot`. The original name will be removed
in a future release. Please update your code. Note that the default `kind` in `factorplo
t` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

Out[121… <seaborn.axisgrid.FacetGrid at 0x2422a765208>

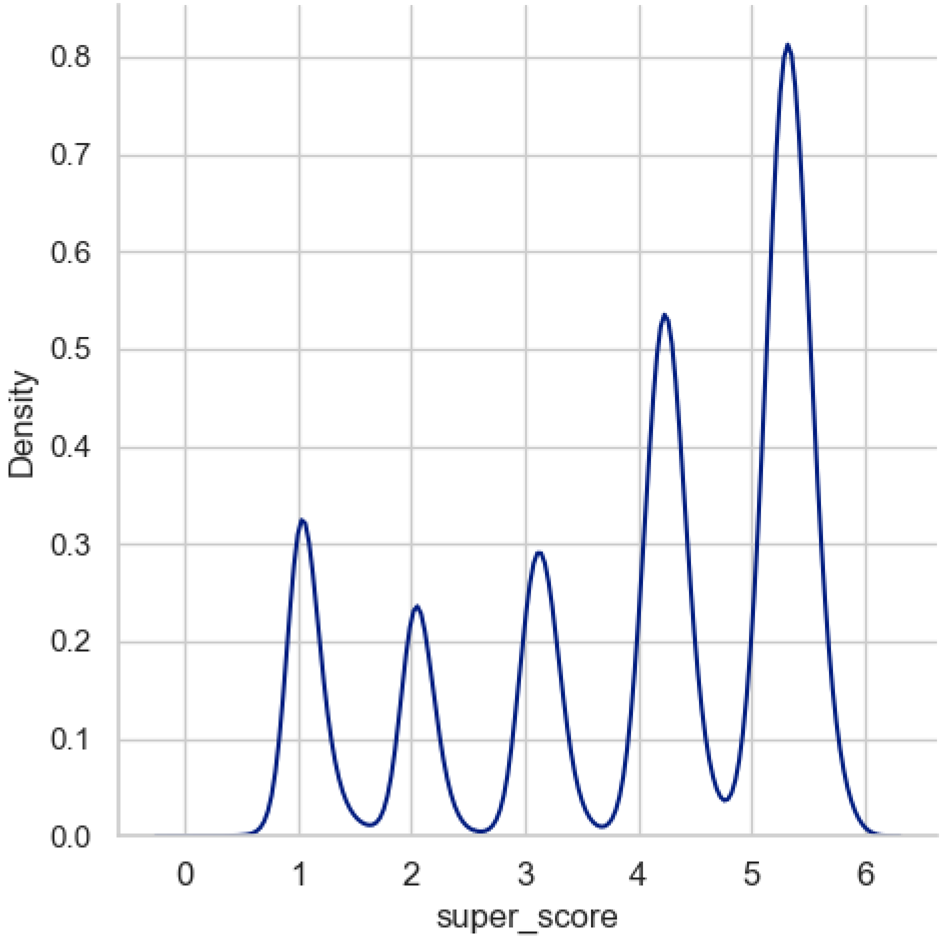


No Concrete inference can be drawn from text_blob sentiment scores as they are more generic

In [122…

```
sns.displot(df_review_sample,
            x="super_score", kind="kde")
```

Out[122… <seaborn.axisgrid.FacetGrid at 0x2422e153a48>



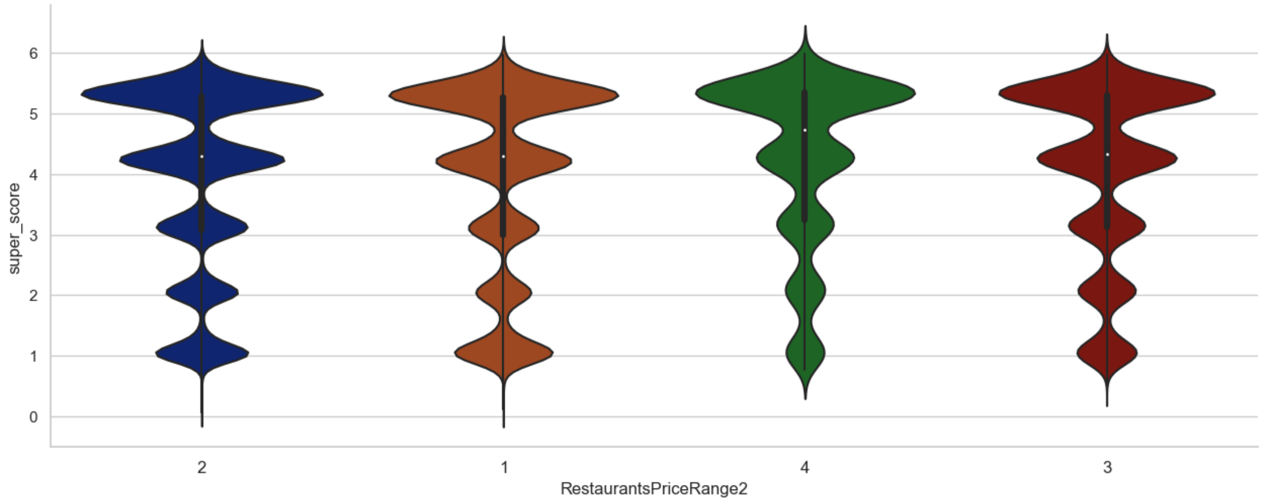
In [123…

```
sns.factorplot(x='RestaurantsPriceRange2',
               y='super_score' ,
               data = df_review_sample ,
               kind='violin', aspect=2.5)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3714: UserWarning: Th
e `factorplot` function has been renamed to `catplot`. The original name will be removed
in a future release. Please update your code. Note that the default `kind` in `factorplo
t` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

Out[123…

```
<seaborn.axisgrid.FacetGrid at 0x2422e03ffc8>
```



Inference similar to price vs stars as seen in the earlier part

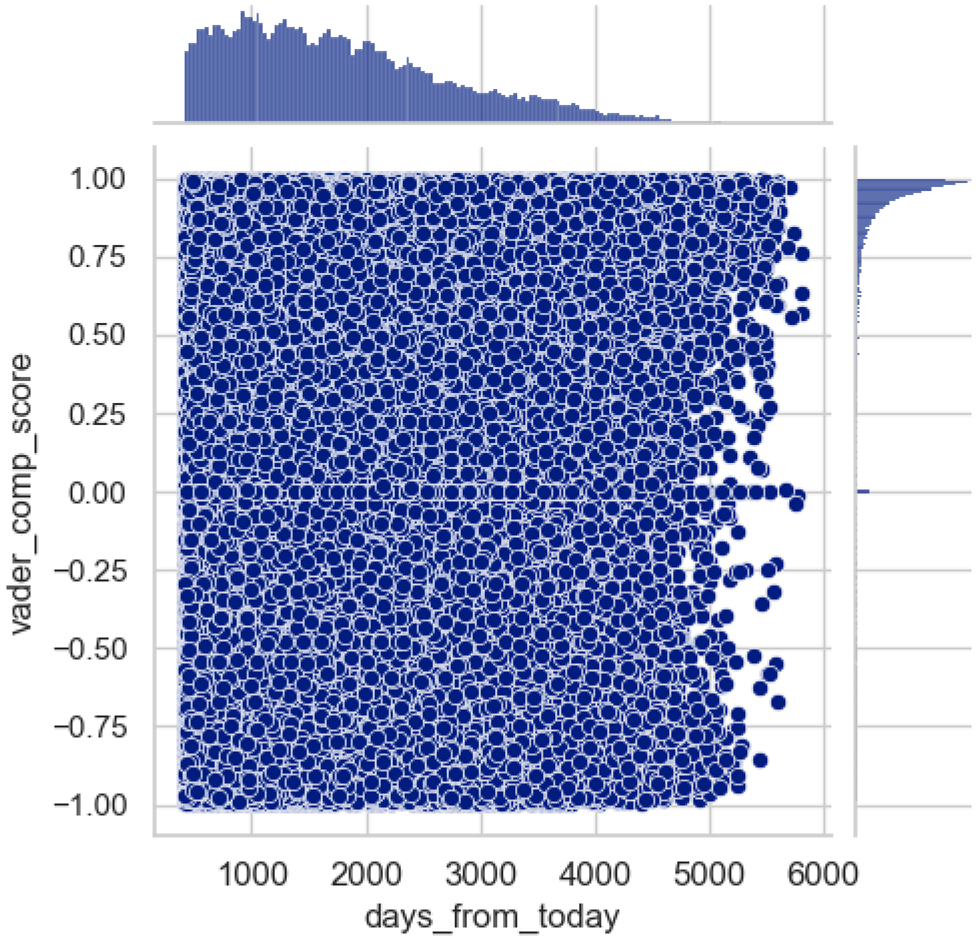
In [124…

```
sns.jointplot(x='days_from_today' , y='vader_comp_score' , data=df_review_sample , size
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\axisgrid.py:2073: UserWarning: The `
size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[124…

```
<seaborn.axisgrid.JointGrid at 0x2422b818b48>
```



No definitve pattern between age of review (as compared to today) and sentiment score

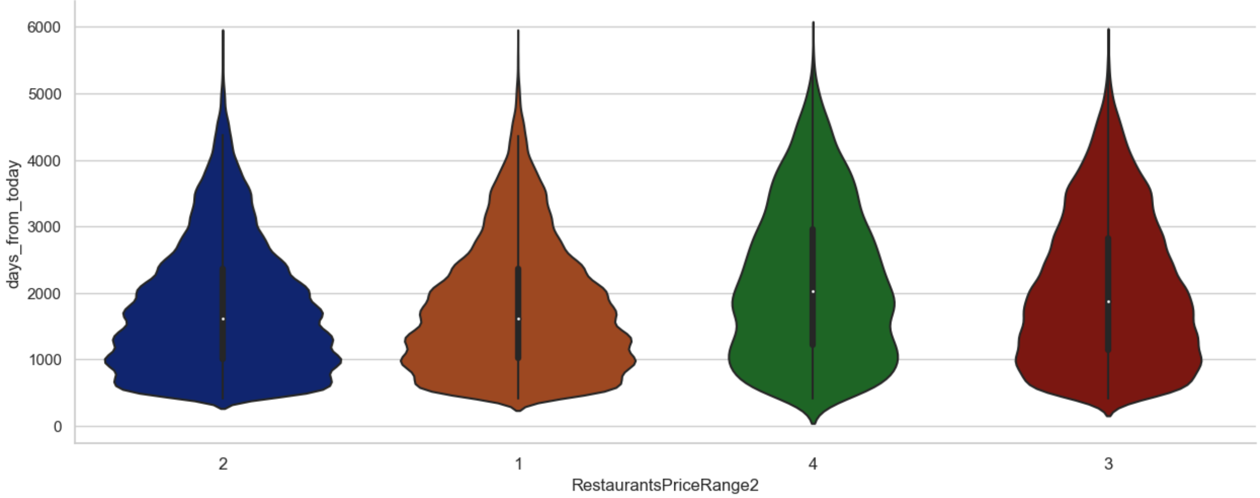
In [125…

```
sns.factorplot(x='RestaurantsPriceRange2',
               y='days_from_today' ,
               data = df_review_sample ,
               kind='violin', aspect=2.5)
```

```
C:\Anaconda\envs\env_yelp\lib\site-packages\seaborn\categorical.py:3714: UserWarning: Th
e `factorplot` function has been renamed to `catplot`. The original name will be removed
in a future release. Please update your code. Note that the default `kind` in `factorplo
t` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

Out[125…

```
<seaborn.axisgrid.FacetGrid at 0x2422e27b6c8>
```



Looks like lower priced restaurants tend to be reviewed more frequently possibly due to sheer qunatum of them in the dataset

In [126…

```
df_review_sample.columns
```

Out[126…

```
Index(['review_id', 'user_id', 'business_id', 'stars', 'useful', 'funny',
       'cool', 'text', 'date', 'city', 'is_open', 'RestaurantsPriceRange2',
       'NoiseLevel', 'CoatCheck', 'Alcohol', 'BusinessAcceptsCreditCards',
       'GoodForKids', 'RestaurantsDelivery', 'RestaurantsTakeOut',
       'OutdoorSeating', 'WiFi', 'Is_chain', 'date1', 'days_from_today',
       'vader_comp_score', 'txt_blb_comp_score', 'super_score'],
      dtype='object')
```

In [127…

```
df_review_sample.to_csv("./data/Review_Business_data_Sample.csv", encoding = 'utf-8')
```

In [ ]: