Natural Language processing for Customer reviews for Hotels

Trip Advisor Location: British Columbia

City Code: 154922

Project Scope: Review positive and negative reviews and attribute them to specific reasons for the reviews to better understand the nature of the hotel and the reason for sentiment.

In [2]:

```
import sys
print(sys.version)
```

```
3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)]
```

In [10]:

```
#Plot
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

#Data Packages
import math
import pandas as pd
import numpy as np

#Progress bar
from tqdm import tqdm

#Counter
from collections import Counter

#Operation
import operator
import csv

#Natural Language Processing Packages
import re
import nltk


## Download Resources
nltk.download("vader_lexicon")
nltk.download("stopwords")
nltk.download("averaged_perceptron_tagger")
nltk.download("wordnet")

from nltk.sentiment import SentimentAnalyzer
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.sentiment.util import *
from nltk import tokenize
from nltk.corpus import stopwords
from nltk.tag import PerceptronTagger
from nltk.data import find

## Machine Learning
import sklearn
import sklearn.metrics as metrics
```

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

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /Users/krutheekarajkumar/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package wordnet to
[nltk_data]     /Users/krutheekarajkumar/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Q1. Sentiment Analysis and Aggregation

NLTK sentiment analysis package (vader) used in conjecture with NLTK library

(a) Computing average Vader sentiment and average ground truth rating per hotel.

In [11]:

```
file = "reviews2.csv"
```

In [12]:

```
with open(file, 'r',encoding='ascii') as f:
    reader = csv.reader(f)
    review = list(reader)
```

In [13]:

```
reviews_df = pd.read_csv(file, names=['Link', 'Hotel_Name', 'Review text', 'Ratings', 'Experience']
, delimiter=',')
#reviews_df.columns('Link','Hotel Name','Review text','Ratings','Experience')
```

In [14]:

```
reviews_df.head()
reviews_df.shape
```

Out[14]:

```
(4059, 5)
```

In [15]:

```
x = reviews_df['Hotel_Name'].unique()
len(x)
# 78 hotels being reviewed
reviews_df.shape
```

Out[15]:

```
(4059, 5)
```

evalSentences :Each of the Reviews were given three scores based on how positive, neutral or negative they were, these were compounded using the NLTK's polarity function. The values were then written into a dataframe which included the text of the review as well as the score associate with it for further analysis

In [18]:

```
def evalSentences(sentences, to_df=False, columns=[2]):
    #Instantiate an instance to access SentimentIntensityAnalyzer class
    print("VADER SENTIMENTS")
    sid = SentimentIntensityAnalyzer()
    pdlist = []
    if to_df:
        for sentence in sentences:
            ss = sid.polarity_scores(sentence)
            pdlist.append([sentence]+[ss['compound']])
        reviewDf = pd.DataFrame(pdlist)
        reviewDf.columns = columns
        return reviewDf
```

```
    else:
        for sentence in sentences:
            #print(sentence)
            ss = sid.polarity_scores(sentence)
            #print(ss)
            for k in sorted(ss):
                print('{0}: {1}, '.format(k, ss[k]), end='')
            print()
```

In [19]:

```
evalSentences(reviews_df['Review text'])
```

```
VADER SENTIMENTS
compound: 0.8442, neg: 0.034, neu: 0.827, pos: 0.14,
compound: 0.9965, neg: 0.011, neu: 0.83, pos: 0.159,
compound: 0.9757, neg: 0.018, neu: 0.739, pos: 0.243,
compound: 0.9766, neg: 0.0, neu: 0.799, pos: 0.201,
compound: 0.4939, neg: 0.019, neu: 0.909, pos: 0.072,
compound: 0.9842, neg: 0.0, neu: 0.763, pos: 0.237,
compound: 0.937, neg: 0.0, neu: 0.796, pos: 0.204,
compound: 0.9692, neg: 0.025, neu: 0.721, pos: 0.254,
compound: 0.996, neg: 0.023, neu: 0.607, pos: 0.37,
compound: 0.9565, neg: 0.0, neu: 0.608, pos: 0.392,
compound: 0.1021, neg: 0.128, neu: 0.74, pos: 0.131,
compound: 0.917, neg: 0.052, neu: 0.702, pos: 0.246,
compound: 0.9392, neg: 0.08, neu: 0.691, pos: 0.229,
compound: 0.9885, neg: 0.023, neu: 0.653, pos: 0.324,
compound: 0.3612, neg: 0.06, neu: 0.851, pos: 0.088,
compound: 0.9694, neg: 0.0, neu: 0.635, pos: 0.365,
compound: 0.9891, neg: 0.0, neu: 0.787, pos: 0.213,
compound: 0.9787, neg: 0.049, neu: 0.601, pos: 0.349,
compound: 0.9432, neg: 0.0, neu: 0.79, pos: 0.21,
compound: -0.7205, neg: 0.096, neu: 0.819, pos: 0.084,
compound: 0.8, neg: 0.097, neu: 0.698, pos: 0.205,
compound: 0.6028, neg: 0.031, neu: 0.895, pos: 0.074,
compound: 0.4915, neg: 0.108, neu: 0.735, pos: 0.157,
compound: 0.7506, neg: 0.099, neu: 0.707, pos: 0.194,
compound: 0.9704, neg: 0.0, neu: 0.543, pos: 0.457,
compound: 0.9942, neg: 0.015, neu: 0.784, pos: 0.201,
compound: 0.9022, neg: 0.09, neu: 0.593, pos: 0.317,
compound: 0.9825, neg: 0.013, neu: 0.844, pos: 0.143,
compound: 0.8682, neg: 0.0, neu: 0.785, pos: 0.215,
compound: 0.9367, neg: 0.009, neu: 0.86, pos: 0.131,
compound: 0.9652, neg: 0.0, neu: 0.744, pos: 0.256,
compound: 0.9803, neg: 0.012, neu: 0.837, pos: 0.151,
compound: 0.9807, neg: 0.0, neu: 0.698, pos: 0.302,
compound: 0.986, neg: 0.0, neu: 0.789, pos: 0.211,
compound: 0.994, neg: 0.046, neu: 0.751, pos: 0.203,
compound: 0.9665, neg: 0.0, neu: 0.809, pos: 0.191,
compound: 0.9443, neg: 0.0, neu: 0.839, pos: 0.161,
compound: 0.9588, neg: 0.0, neu: 0.626, pos: 0.374,
compound: 0.9638, neg: 0.0, neu: 0.798, pos: 0.202,
compound: 0.8962, neg: 0.07, neu: 0.79, pos: 0.141,
compound: 0.8551, neg: 0.056, neu: 0.702, pos: 0.243,
compound: 0.9273, neg: 0.043, neu: 0.712, pos: 0.245,
compound: 0.9716, neg: 0.0, neu: 0.672, pos: 0.328,
compound: 0.8937, neg: 0.0, neu: 0.749, pos: 0.251,
compound: 0.9757, neg: 0.029, neu: 0.773, pos: 0.198,
compound: 0.9864, neg: 0.025, neu: 0.582, pos: 0.393,
compound: 0.9532, neg: 0.0, neu: 0.758, pos: 0.242,
compound: 0.9576, neg: 0.0, neu: 0.679, pos: 0.321,
compound: 0.9466, neg: 0.0, neu: 0.871, pos: 0.129,
compound: 0.9744, neg: 0.0, neu: 0.809, pos: 0.191,
compound: 0.9836, neg: 0.011, neu: 0.775, pos: 0.214,
compound: 0.9718, neg: 0.0, neu: 0.617, pos: 0.383,
compound: 0.9895, neg: 0.0, neu: 0.707, pos: 0.293,
compound: 0.9897, neg: 0.0, neu: 0.537, pos: 0.463,
compound: 0.8761, neg: 0.0, neu: 0.829, pos: 0.171,
compound: 0.9545, neg: 0.0, neu: 0.637, pos: 0.363,
compound: 0.97, neg: 0.0, neu: 0.642, pos: 0.358,
compound: 0.9881, neg: 0.0, neu: 0.763, pos: 0.237,
compound: 0.9932, neg: 0.0, neu: 0.662, pos: 0.338,
compound: 0.968, neg: 0.0, neu: 0.81, pos: 0.19,
compound: 0.9545, neg: 0.04, neu: 0.62, pos: 0.339,
```

In [20]:

```
reviews = reviews_df['Review text'].values
reviewDF = evalSentences(reviews, to_df=True, columns=['reviewCol','vader'])
```

VADER SENTIMENTS

In [21]:

```
x = reviews_df["Experience"]
y = reviews_df['Hotel_Name']
reviewDF = pd.concat([reviewDF, x], axis=1)
reviewDF = pd.concat([reviewDF, y], axis=1)
```

In [22]:

```
reviewDF[reviewDF['Hotel_Name'] == 'Stansbury&#39;s Guest House']
```

Out[22]:

	reviewCol	vader	Experience	Hotel_Name
28	"Stayed for a week in the Stansbury\\'s guest ...	0.8682	positive	Stansbury's Guest House
29	"A group of 6 of us stayed here for a weekend ...	0.9367	positive	Stansbury's Guest House
30	"Visiting from New Zealand,we were fortunate t...	0.9652	positive	Stansbury's Guest House
31	"Gwen and Scott have a beautiful Guest House a...	0.9803	positive	Stansbury's Guest House
32	"A beautifully finished apartment with every c...	0.9807	positive	Stansbury's Guest House
33	"We happended upon Stansbury\\'s last year jus...	0.9860	positive	Stansbury's Guest House
34	"We stayed at Stansbury`s Guest House for 8 da...	0.9940	positive	Stansbury's Guest House
35	"We had an amazing stay at the Stansbury\\'s G...	0.9665	positive	Stansbury's Guest House
36	"Nine of us spent the weekend at the Stansbury...	0.9443	positive	Stansbury's Guest House
37	"This place has: very friendly hosts, is super...	0.9588	positive	Stansbury's Guest House
38	"Cumberland is cute little village in the Como...	0.9638	positive	Stansbury's Guest House
39	"My wife and I stayed at Stansbury\\'s Guest H...	0.8962	positive	Stansbury's Guest House
40	"You will not be disappointed! The rooms and ...	0.8551	positive	Stansbury's Guest House
41	"What a wonderful Guest House to call home for...	0.9273	positive	Stansbury's Guest House
42	"This house was clean and had a beautiful sunn...	0.9716	positive	Stansbury's Guest House
43	"My sister and I just spent 10 days at the Gue...	0.8937	positive	Stansbury's Guest House
44	"I hardly want to share this gem so that Gwen\...	0.9757	positive	Stansbury's Guest House
45	"What a great place! 1, 2, or 3 bedrooms. Bi...	0.9864	positive	Stansbury's Guest House
46	"Right from the beginning Stansbury\\'s Guest ...	0.9532	positive	Stansbury's Guest House
47	"Gwen and Scott.\\n\\nOur stay was Fantastic!!...	0.9576	positive	Stansbury's Guest House
48	"My husband and I stayed at Stansbury\\'s with...	0.9466	positive	Stansbury's Guest House
49	"We have stayed at Stansbury\\'s Guest House a...	0.9744	positive	Stansbury's Guest House
50	"I finally made my way to Cumberland and at th...	0.9836	positive	Stansbury's Guest House
51	"Great location! Close to trails, pubs, food a...	0.9718	positive	Stansbury's Guest House
52	"My Husband and I came for a mountain bike vac...	0.9895	positive	Stansbury's Guest House
53	"Gwen & Scott have created a wonderful san...	0.9897	positive	Stansbury's Guest House
54	"Scott and Gwen made us very welcome. I rode w...	0.8761	positive	Stansbury's Guest House
55	"The hosts are warm and inviting and the 3 bed...	0.9545	positive	Stansbury's Guest House
56	"A perfect place to stay if you want a peacefu...	0.9700	positive	Stansbury's Guest House
57	"Always on the lookout for interesting places ...	0.9881	positive	Stansbury's Guest House
...	...	...	...	...
59	"My son and I spent three days in Cumberland m...	0.9680	positive	Stansbury's Guest House

	reviewCol	vader	Experience	Hotel_Name
60	"Scott and Gwen have it figured out! Provided...	0.9545	positive	Stansbury's Guest House
61	"Great location, easy walk to Main Street of C...	0.9638	positive	Stansbury's Guest House
62	"Cute modern decor. I love the headboard in th...	0.9749	positive	Stansbury's Guest House
63	"Anyone finding themselves up island from Vict...	0.9167	positive	Stansbury's Guest House
64	"This 3 bedroom suite was everything you want ...	0.9712	positive	Stansbury's Guest House
65	"Great place to stay with superb hosts. It pro...	0.9298	positive	Stansbury's Guest House
66	"Gwen and Scott are lovely friendly people! Gw...	0.9824	positive	Stansbury's Guest House
67	"We stayed here for 6 weeks while our new home...	0.8807	positive	Stansbury's Guest House
68	"Margaret and I stayed for a couple of nights ...	0.9147	positive	Stansbury's Guest House
69	"Stansbury\\'s is a warm, inviting place to st...	0.9880	positive	Stansbury's Guest House
70	"Gwen and Scott have a clean, well run accommo...	0.9136	positive	Stansbury's Guest House
71	"Couldn\\'t be a nicer location and the accomm...	0.9184	positive	Stansbury's Guest House
72	"My partner stayed at Stansbury\\'s Guest Hous...	0.9885	positive	Stansbury's Guest House
73	"It is a well run Guest House. Room was very c...	0.9335	positive	Stansbury's Guest House
74	"We loved the decor of the suite and full amen...	0.9584	positive	Stansbury's Guest House
75	"Stansbury\\'s is a perfect place in so many w...	0.9496	positive	Stansbury's Guest House
76	"The owners are nice as is the property and ro...	0.8905	positive	Stansbury's Guest House
77	"We enjoyed our stay at Stansbury\\'s. The roo...	0.8646	positive	Stansbury's Guest House
78	"Stansbury\\'s Guest House was a perfect locat...	0.8519	positive	Stansbury's Guest House
79	"A lovely welcoming place to stay. Spacious, v...	0.9745	positive	Stansbury's Guest House
80	"We stayed in the 3 bedroom unit. It\\'s perfe...	0.9818	positive	Stansbury's Guest House
81	"Chance meeting Gwen in July walking around Cu...	0.9570	positive	Stansbury's Guest House
82	"I\\'ve stayed at Stansbury\\'s Guest House tw...	0.9501	positive	Stansbury's Guest House
83	"We had a very nice stay. Scott and Gwen are v...	0.9376	positive	Stansbury's Guest House
84	"We stayed here for the Woodstove Music festiv...	0.9881	positive	Stansbury's Guest House
85	"Our suite had everything we needed. This is a...	0.9628	positive	Stansbury's Guest House
86	"Thanks to Gwen for taking care of us (even br...	0.8550	positive	Stansbury's Guest House
87	"We booked the three bedroom suite at Stansbur...	0.9734	positive	Stansbury's Guest House
88	"This guest house is awesome. It is run person...	0.9895	positive	Stansbury's Guest House

61 rows × 4 columns

(b) Rank hotels by

(i) Average Ground Truth Sentiment
(ii) Average Vader Compound Sentiment Score

New Dataframe created with exp_index set as a binary column to suggest if the over all sentiment was positive (1) or negative (0)

In [23]:

```
exp = reviewDF['Experience']
exp_list = []
for e in exp:
    if e == 'positive':
        exp_list.append(1)
    else:
        exp_list.append(0)
```

In [24]:

```
exp_df = pd.DataFrame(exp_list,columns=['exp_index'])
reviewDF = pd.concat([reviewDF, exp_df], axis=1)
```

In [25]:

```
reviewDF.head()
```

Out[25]:

	reviewCol	vader	Experience	Hotel_Name	exp_index
0	"I was looking for a place to sit and chill fo...	0.8442	positive	The Riding Fool Hostel	1
1	"I stayed at the Riding Fool Hostel whilst I w...	0.9965	positive	The Riding Fool Hostel	1
2	"My husband and I (both in our 50\\'s) stayed ...	0.9757	positive	The Riding Fool Hostel	1
3	"We were warmly welcomed by Caitlin, the new ...	0.9766	positive	The Riding Fool Hostel	1
4	"Comfortable, cheap, and cosy accommodation wi...	0.4939	positive	The Riding Fool Hostel	1

In [26]:

```
# Grouping by hotel name to understand the average review of each of the hotel and the number of r
eviews for them
# The "reviewCol" column is used to denote the number of reviews for each of the hotels
avg_rating = reviewDF.groupby('Hotel_Name').mean()
count = reviewDF.groupby('Hotel_Name').count()
print("The average vader rating grouped by hotel is: ",avg_rating.head())
print("The number of reviews grouped by hotels are: ", count.head())
```

```
The average vader rating grouped by hotel is:                            vader  exp_index
Hotel_Name
A&amp;J B&amp;B          0.973442   1.000000
A-1 Alberni Inn          0.447724   0.523810
Abbotsford Hotel         0.225563   0.359375
Arbutus Grove Motel      0.897362   0.938144
Artful Retreat B&amp;B   0.888300   1.000000
The number of reviews grouped by hotels are:                          reviewCol  vader  Experience
exp_index
Hotel_Name
A&amp;J B&amp;B                 19      19          19          19
A-1 Alberni Inn                21      21          21          21
Abbotsford Hotel               64      64          64          64
Arbutus Grove Motel            97      97          97          97
Artful Retreat B&amp;B          1       1           1           1
```

In [27]:

```
temp = count['reviewCol']
avg_rating = pd.concat([avg_rating, temp], axis=1)
avg_rating.head()
```

Out[27]:

Hotel_Name	vader	exp_index	reviewCol
A&J B&B	0.973442	1.000000	19
A-1 Alberni Inn	0.447724	0.523810	21
Abbotsford Hotel	0.225563	0.359375	64
Arbutus Grove Motel	0.897362	0.938144	97
Artful Retreat B&B	0.888300	1.000000	1

In [28]:

```
hotels_sorted = avg_rating.sort_values(by =["exp_index","reviewCol"],ascending=False)
print("TOP 5 Ranking hotels by Average Ground Truth ")
```

```
top_5_GT = hotels_sorted.head(22)
top_5_GT.head()
```

```
TOP 5 Ranking hotels by Average Ground Truth
```

Out[28]:

Hotel_Name	vader	exp_index	reviewCol
Stansbury's Guest House	0.948890	1.0	61
Cedar Song B&B and Cottage	0.942706	1.0	49
Stamp Falls B & B	0.938477	1.0	26
Mozey-On-Inn	0.913273	1.0	22
Nimpo Lake Resort	0.940255	1.0	22

In [29]:

```
hotels_sorted_groundTruth = avg_rating.sort_values(by ="exp_index",ascending=True)
print("BOTTOM 5 Ranking hotels by Average Ground Truth ")
hotels_sorted_groundTruth.head()
```

```
BOTTOM 5 Ranking hotels by Average Ground Truth
```

Out[29]:

Hotel_Name	vader	exp_index	reviewCol
Howard Johnson Hotel Port Alberni	0.405518	0.243902	82
Sorrento Inn	0.241462	0.250000	16
Bluebird Motel	0.385882	0.272727	11
Deerview Lodge & Cabins	0.471357	0.285714	7
Tyee Village Motel	0.322247	0.333333	15

In [30]:

```
hotels_sorted_by_vader = avg_rating.sort_values(by ="vader",ascending=True)
print("BOTTOM 5 Ranking hotels by Vader Compound Sentiment")
hotels_sorted_by_vader.head()
```

```
BOTTOM 5 Ranking hotels by Vader Compound Sentiment
```

Out[30]:

Hotel_Name	vader	exp_index	reviewCol
Jewel Bay Resort	0.205180	0.400000	5
Abbotsford Hotel	0.225563	0.359375	64
Sorrento Inn	0.241462	0.250000	16
Tyee Village Motel	0.322247	0.333333	15
China Creek Campground	0.335480	0.400000	10

In [31]:

```
hotels_sorted = avg_rating.sort_values(by =["vader"],ascending=False)
print("TOP 5 Ranking hotels by Vader")
```

```
hotels_sorted.head()
```

```
TOP 5 Ranking hotels by Vader
```

Out[31]:

Hotel_Name	vader	exp_index	reviewCol
A&J B&B	0.973442	1.0	19
Retreat Wilderness Inn	0.972945	1.0	11
Mt H'Kusam View Lodge	0.968571	1.0	7
San Jose River Ranch Cariboo B&B	0.967820	1.0	5
Char's Landing Guesthouse	0.967400	1.0	2

Comparison of Top 5 hotels based on average ratings vs average vader scores:

Top 5 according to average ground truth: Stansbury's Guest House, Cedar Song B&B and Cottage, Stamp Falls B, Mozey-On-Inn, Nimpo Lake Resort
Top 5 according to average vader: A&J B&B, Retreat Wilderness Inn, Mt H'Kusam View Lodge, San Jose River Ranch Cariboo B&B, Char's Landing Guesthouse
The top five hotels for the average vader score and the average ground truth scores do not match as there are 22 hotels that have a perfect average ground truth rating (100% of the people visiting had positive experiences). The degree to which they had a positive experience is not captures by the binary rating. However, it is captured by the vader scores. The top five hotels based on average ground truth was first filtered as having all positive (1.0) and then filtered by the number of reviews for that hotel. This is why the top five from vader scores do not apprear in the top five of the ground truth average.

Q2. Frequency Analysis

(a) Using term frequency of the words for :

(i) positive reviews and
(ii) negative with ground truth sentiment to rank the top-50 most frequent non-stopwords in the review collection.

In [32]:

```
def freq_analysis(reviews, k):
    stop = set(stopwords.words('english'))
    counter = Counter()
    for review in reviews:
            counter.update([word.lower()
                            for word
                            in re.findall(r'\w+', review)
                            if word.lower() not in stop and len(word) > 2])
    topk = counter.most_common(k)
    return topk
```

In [33]:

```
print("Top 50 words used in the positive reviews")
# Extracting only the positivly reviewed hotels and their associated text to study the most freque
ntly used words
pos = reviews_df[reviews_df["Experience"]=='positive']['Review text']
result_pos = freq_analysis(pos,50)
l_pos = len(pos)-1
result_pos
```

```
Top 50 words used in the positive reviews
```

Out[33]:

```
[('room', 2391),
 ('great', 1980),
 ('stay', 1830),
```

```
 ('clean', 1586),
 ('hotel', 1578),
 ('staff', 1538),
 ('beach', 1265),
 ('stayed', 1181),
 ('good', 1163),
 ('well', 1146),
 ('would', 1135),
 ('nice', 1124),
 ('friendly', 1113),
 ('comfortable', 1069),
 ('one', 1034),
 ('place', 1023),
 ('rooms', 957),
 ('breakfast', 904),
 ('resort', 844),
 ('time', 763),
 ('restaurant', 744),
 ('area', 719),
 ('night', 683),
 ('back', 678),
 ('pool', 667),
 ('helpful', 646),
 ('view', 643),
 ('service', 623),
 ('location', 598),
 ('also', 576),
 ('family', 576),
 ('nthe', 571),
 ('bed', 564),
 ('two', 559),
 ('food', 545),
 ('amp', 532),
 ('recommend', 520),
 ('like', 519),
 ('beautiful', 518),
 ('day', 516),
 ('really', 505),
 ('quiet', 503),
 ('excellent', 497),
 ('get', 496),
 ('best', 493),
 ('front', 493),
 ('everything', 491),
 ('kitchen', 478),
 ('beds', 474),
 ('could', 461)]
```

In [34]:

```
pos_df = pd.DataFrame({'Index':pos.index, 'Review':pos.values})
pos_df[3354:3358]
pos_rev_df= pos_df['Review']
pos_rev_df.head()
```

Out[34]:

```
0    "I was looking for a place to sit and chill fo...
1    "I stayed at the Riding Fool Hostel whilst I w...
2    "My husband and I (both in our 50\\'s) stayed ...
3    "We were warmly  welcomed by Caitlin, the new ...
4    "Comfortable, cheap, and cosy accommodation wi...
Name: Review, dtype: object
```

In [35]:

```
print("Top 50 words used in the negative reviews")
neg = reviews_df[reviews_df["Experience"]=='negative']['Review text']
result_neg = freq_analysis(neg,50)
result_neg
#len(neg)
```

```
Top 50 words used in the negative reviews
```

Out[35]:

```
[('room', 1054),
 ('hotel', 565),
 ('would', 405),
 ('stay', 368),
 ('one', 318),
 ('night', 302),
 ('quot', 302),
 ('clean', 288),
 ('good', 277),
 ('place', 275),
 ('rooms', 265),
 ('staff', 250),
 ('nice', 239),
 ('breakfast', 225),
 ('stayed', 217),
 ('time', 211),
 ('get', 201),
 ('desk', 196),
 ('bed', 189),
 ('nthe', 185),
 ('front', 184),
 ('like', 176),
 ('could', 173),
 ('great', 164),
 ('even', 157),
 ('bathroom', 154),
 ('back', 153),
 ('well', 148),
 ('restaurant', 146),
 ('check', 145),
 ('small', 143),
 ('door', 142),
 ('beach', 142),
 ('day', 141),
 ('resort', 141),
 ('service', 141),
 ('next', 140),
 ('also', 138),
 ('friendly', 132),
 ('motel', 132),
 ('two', 131),
 ('got', 126),
 ('floor', 125),
 ('told', 124),
 ('really', 122),
 ('little', 122),
 ('pool', 120),
 ('much', 119),
 ('view', 118),
 ('said', 115)]
```

In [36]:

```
words_pos = [i[0] for i in result_pos]
words_neg = [i[0] for i in result_neg]
```

In [37]:

```
#Some words are repeating themselves and it is useful to see which words were common for both list
s
s = set(words_neg)
common_words = [x for x in words_pos if x in s]
common_words
```

Out[37]:

```
['room',
 'great',
 'stay',
 'clean',
 'hotel',
 'staff',
 'beach',
```

```
 stayed ,
'good',
'well',
'would',
'nice',
'friendly',
'one',
'place',
'rooms',
'breakfast',
'resort',
'time',
'restaurant',
'night',
'back',
'pool',
'view',
'service',
'also',
'nthe',
'bed',
'two',
'like',
'day',
'really',
'get',
'front',
'could']
```

In [38]:

```
count = 0
for review in neg:
    if 'great' in review:
        count += 1
        print (count,"Review number",review)
        print("")
```

```
1 Review number "My biggest complaint with the Riding Fool is the lack of door closure on the dorm room doors. Resulting in the door never properly closing, or banging shut, both creating more nois e in the room than necessary. Bunks are stable so you don\\'t really hear the top one going to bed which is great. Lockers are a good size. Kitchen is well equipped.

2 Review number "Stayed in Room 7 recently. Noise from the staff quarters immediately below made s leeping difficult. The whole establishment is spacious, airy, super-clean, and the facilities are outstanding. And Cumberland is a great place to relax. But the sound-proofing in the hostel is ter rible. Take your ear plugs.

3 Review number "5 of us ladies (and no, we are not miners) were in Sparwood for a night and we st ayed at this hotel.  The staff was great, the bed was comfortable, and the hotel was clean.  The r oom had a fridge, coffee maker and microwave.  We had breakfast at the restaurant.  Food was fine, and the servings were huge.\\n\\nNone of us would have anything bad to say about the hotel.

4 Review number "Being that this was the only available hotel along our route (we didn\\'t realise Fernie was so close), it was our only option. \\n\\nThe hotel is dated looking, both inside and ou t. \\nThe room was sub-standard for the price we paid. The beds were not terribly comfy and the fl oor had many dark stains in the carpet. That being said, there was a small bar fridge and a coffee maker. The bathroom was tiny and I hope clean. There was a ring around the tub in the bath that I did not notice until my first child was done with a bath (I can honestly say he wasn\\'t dirty eno ugh to cause that, and a properly cleaned bath would not have a ring appear so quickly if my child was that dirty). \\n\\nThe restaurant was great though! We had breakfast there and the food was gr eat, fresh and very filling. It was reasonably priced for a sit down place. The service was most e xcellent and we were treated very nicely. \\n\\nOn the whole, I would be very unlikely to stay the re again (would rather stay in Fernie, 20 minutes away). I would however eat at that restaurant ag ain next time I am passing through.

5 Review number "I recently stayed at the Causeway for a ball tournament with my daughter.  Althou gh our hotel room was fairly clean, the foyer carpet was nasty, as were the hallways and stairways .  \\n\\nThe staff at the hotel was courteous and friendly.  The housekeepers did a great job.\\n\\nThe restaurant staff were very slow and lacked any enthusiasm whatsoever.  By my count, only 3 other tables were being occupied at the time and our breakfast took over 45 mins to receive .  The food was average and for breakfast, it was fairly pricey.

6 Review number "Stayed here for two nights to save money as opposed to paying higher prices in fe rnie.  Was shocked at how nice it was for the price.  Rooms are a good size with everything you ne ed fridge microwave ....  Beds aren\\'t the greatest but overall pleasantly surprised.  Would sta
```

he motel to be on the beach, or at least have a view of the beach, the access is a short walk. The beach is much less populated than the community beach which is just on the other side of the bay.\\n\\nI rated the sleep quality poor because the walls are on the paper thin side and the couple next door were up all night. The management could not be reached and didn\\'t ask them to leave the next day. I moved rooms to not have to listen to the racket all the next night.

117 Review number "Great location, great price, close to everything. The motel is older but is nice and clean, comes with a free game of mini golf. But.......be warned, there is no air conditioning. We went the weekend of August 2 and it was very hot out. We were warned that there was no air, but never expected it to be as uncomfortably hot in the room as it was. We did not sleep the night we were there, spent most of the night in a chair in front of the door trying to get any bit of breeze that might be coming by. There was a ceiling fan and an oscillating fan but it didn\\'t seem to help much. We were promised a main floor room which I think would have been a little cooler than the second floor. Somehow our room got moved up there. :(The only little ice machine is located inside the office that said it closed at 11:00pm. I went down to stock up on ice knowing I wouldn\\'t be sleeping much and it had closed 10 minutes before 11:00. Needless to say there were several disappointed guests. Next year we will be booking at a hotel/motel with air conditioning.

118 Review number "The hotel was in a great location and was comfortable enough. I didn\\'t realize there was no air conditioning when I booked it so that was my own fault.\\n\\nIt is a pet friendly hotel so that was great for us as we had our puppy with us.\\n\\nThe wifi was extremely slow and breakfast wasn\\'t until 8am and we had to leave before then.\\n\\nIt wouldn\\'t be our first choice for our next stay in Parksville.

119 Review number "This was a nice, older, but clean motel with very comfortable beds, great rates, pet friendly, close to ocean, breakfast included along with passes to the Mini Golf next door and close enough to restaurants that you can walk to. No air conditioning in our room though, which made for a not so comfortable sleep.

120 Review number "Paradise Seashell Motel is in an excellent location to visit Coombs (goats on the roof ice cream and very cool statues), drive to west side of Island, see the amazing sand sculptures, walk on a very nice Boardwalk, and of course, go to the beach. The mini golf next door is owned by Paradise, and you can get the golf free, although we never did play. It looked like a very nice "course" and there are bumper boats too.\\nNow , the rooms. Having read the reviews, we were disappointed to find that not only were we upstairs and had to drag our golf clubs and bags up the steps (We should have probably requested downstairs at booking time as we are not young except at heart), but our room was minimal. We had a comfortable queen bed, a desk, a coffee maker and coffee packets, a tv and dresser and closet. We also had an extra chair and fan by the locked connecting door. We could hear every word being said through the door on our first night, but it was quiet the next two nights. As it turns out, we peeked in a room on our way to breakfast, and discovered we were in the second bedroom of a large suite with two queen beds and a kitchen. So, while our room was adequate for us, if one travels with a family, be sure to ask for the front rooms or entire suite. Barbecues and picnic tables are on a grassy area, perfect for families who don\\'t want the expense of eating out a lot. The two big pluses are the breakfast and the staff. Breakfast is served from 7:30 to , I think, 10. Hard boiled eggs, bagels, muffins, toast, cereal, fruit and great coffee can be eaten in the dining room or at outside tables. The staff is friendly and very attentive. A wand fell off my blinds; fixed next day while we toured. I requested extra soap and shampoo, found it in bathroom on our return. The gal behind the desk on checking was even going to have a staff member haul our golf clubs up the stairs for us. It was certainly a nice offer, even if we didn\\'t opt to take it. Oh, and the wifi was very good. Overall, it is a comfortable place to stay\\n\\nOverall, I would recommend

121 Review number "Stayed here this year with Grandma and the little nieces. The location was great being right next to Mini golf, bumper boats etc.. The price was unbeatable especially with a simple breakfast included (coffee, tea, juice, cereals, fruit, bread, muffins and english muffins). As well as a free found of mini golf for each of us. There are restaurants near by and the beach and playground are a short drive away. The beds in our room were excellent, we also had a small kitchenette with a fridge and microwave which was unexpected. The rooms are dark and fairly dated but are clean and have been updated with a few modern fixes. So honestly considering the price it was great! I don\\'t recommend it if you are looking for something new and modern.

Position and usage of the top-ranked words:

There were multiple words that were present in both the negative review and the positive review. While most of these words were neutral nouns, the adjectives attached to the words would be the emotion carrying word (the context set the tone of the word). However, words such as "great" and "clean" appear in the negative review, this is because one part of the review is positive in nature, however the core of the review is still negative. This is evident by vitual inspection from the previous cell, where certain aspects of the hotel was "great", while other parts were bad and therefore catagorized as a negative ground truth.

In [39]:

```
#Analyzing the context by implmenting Part-of-speech tagging of full review sentences
tagger = PerceptronTagger()
taggedToks=[]
pos_tag = tagger.tag
for i in range(l_pos):
    t=i+1
    #print(pos_df[i:t])
    taggedToks.append(pos_tag(re.findall(r'\w+', pos_rev_df[i])))
```

In [40]:

```
taggedToks[:10]
```

Out[40]:

```
[[('I', 'PRP'),
  ('was', 'VBD'),
  ('looking', 'VBG'),
  ('for', 'IN'),
  ('a', 'DT'),
  ('place', 'NN'),
  ('to', 'TO'),
  ('sit', 'VB'),
  ('and', 'CC'),
  ('chill', 'VB'),
  ('for', 'IN'),
  ('a', 'DT'),
  ('while', 'NN'),
  ('and', 'CC'),
  ('I', 'PRP'),
  ('found', 'VBD'),
  ('it', 'PRP'),
  ('in', 'IN'),
  ('the', 'DT'),
  ('Riding', 'NNP'),
  ('Fool', 'NNP'),
  ('Hostel', 'NNP'),
  ('in', 'IN'),
  ('Cumberland', 'NNP'),
  ('Great', 'NNP'),
  ('kitchen', 'NN'),
  ('and', 'CC'),
  ('common', 'JJ'),
  ('area', 'NN'),
  ('comfy', 'NN'),
  ('beds', 'NNS'),
  ('and', 'CC'),
  ('nice', 'JJ'),
  ('folks', 'NNS'),
  ('What', 'WP'),
  ('more', 'JJR'),
  ('could', 'MD'),
  ('you', 'PRP'),
  ('ask', 'VB'),
  ('for', 'IN'),
  ('Oh', 'NNP'),
  ('there', 'EX'),
  ('s', 'VBZ'),
  ('the', 'DT'),
  ('nearby', 'JJ'),
  ('mountain', 'NN'),
  ('biking', 'NN'),
  ('trails', 'VBZ'),
  ('rivers', 'NNS'),
  ('lakes', 'VBZ'),
  ('not', 'RB'),
  ('to', 'TO'),
  ('mention', 'VB'),
  ('the', 'DT'),
  ('big', 'JJ'),
  ('mountain', 'NN'),
  ('just', 'RB'),
  ('a', 'DT'),
  ('short', 'JJ'),
  ('distance', 'NN'),
  ('away', 'RB'),
```

```
  ('here', 'RB'),
  ('The', 'DT'),
  ('hostel', 'NN'),
  ('has', 'VBZ'),
  ('a', 'DT'),
  ('great', 'JJ'),
  ('atmosphere', 'NN'),
  ('and', 'CC'),
  ('a', 'DT'),
  ('certain', 'JJ'),
  ('charm', 'NN'),
  ('nIt', 'NN'),
  ('is', 'VBZ'),
  ('a', 'DT'),
  ('great', 'JJ'),
  ('place', 'NN'),
  ('for', 'IN'),
  ('mountain', 'NN'),
  ('bikers', 'NNS'),
  ('nAlthough', 'IN'),
  ('I', 'PRP'),
  ('didn', 'VBP'),
  ('t', 'NNS'),
  ('have', 'VBP'),
  ('kids', 'NNS'),
  ('with', 'IN'),
  ('my', 'PRP$'),
  ('I', 'PRP'),
  ('think', 'VBP'),
  ('it', 'PRP'),
  ('would', 'MD'),
  ('be', 'VB'),
  ('a', 'DT'),
  ('fine', 'JJ'),
  ('environment', 'NN'),
  ('for', 'IN'),
  ('families', 'NNS'),
  ('too', 'RB')]]
```

In [41]:

```
def flatten(npTokenList):
    finalList =[]
    for phrase in npTokenList:
        token = ''
        for word in phrase:
            token += word + ' '
        finalList.append(token.rstrip())
    return finalList
```

In [42]:

```
import itertools
```

In [43]:

```
len(taggedToks)
taggedToks[43:5]
#toks_flat = taggedToks.flat()
merged = list(itertools.chain(*taggedToks))
merged[:4]
```

Out[43]:

```
[('I', 'PRP'), ('was', 'VBD'), ('looking', 'VBG'), ('for', 'IN')]
```

In [44]:

```
# Defining the chunkstring
grammar = r"""
    NBAR:
        {<NN.*|JJ>*<NN.*>}  # Nouns and Adjectives, terminated with Nouns
```

```
    NP:
        {<NBAR>}
        {<NBAR><IN><NBAR>}  # Above, connected with in/of/etc...
"""
```

In [45]:

```
# The behaviour of the parser is specified to "grammar" in order to add more structure to the sent
ence
# The defined parser uses the POS tagged reviews as input
chunker = nltk.RegexpParser(grammar)
tree= chunker.parse(merged)
```

In [46]:

```
# Noun Phrase Extraction Support Functions
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
# Defining both lemmatizing and stemming - although since lemmatization uses the context to conver
the word into the
# base form - it is predicted to have more value
lemmatizer = nltk.WordNetLemmatizer()
stemmer = nltk.stem.porter.PorterStemmer()

# generator, generate leaves one by one
def leaves(tree):
    """Finds NP (nounphrase) leaf nodes of a chunk tree."""
    for subtree in tree.subtrees(filter = lambda t: t.label()=='NP' or t.label()=='JJ' or t.label()
=='RB'):
        yield subtree.leaves()

# stemming, lematizing, lower case
def normalise(word):
    """Normalises words to lowercase and stems and lemmatizes it."""
    word = word.lower()
    word = stemmer.stem(word)
    word = lemmatizer.lemmatize(word)
    return word

# stop-words and length control
def acceptable_word(word):
    """Checks conditions for acceptable word: length, stopword."""
    accepted = bool(2 <= len(word) <= 400
        and word.lower() not in stop_words)
    return accepted

# generator, create item once a time
def get_terms(tree):
    for leaf in leaves(tree):
        term = [normalise(w) for w,t in leaf if acceptable_word(w) ]
        # Phrase only
        if len(term)>1:
            yield term
```

In [56]:

```
npTokenList = [word for word in get_terms(tree)]
npTokenList[:10]
```

Out[56]:

```
[['ride', 'fool', 'hostel'],
 ['cumberland', 'great', 'kitchen'],
 ['common', 'area', 'comfi', 'bed'],
 ['nice', 'folk'],
 ['nearbi', 'mountain', 'bike'],
 ['big', 'mountain'],
 ['short', 'distanc'],
 ['surpris', 'amount'],
 ['ride', 'fool', 'hostel', 'whilst'],
 ['mount', 'washington', 'alpin', 'resort']]
```

In [48]:

In [48]:

```
finalList = flatten(npTokenList)
```

In [49]:

```
(finalList[:6])
```

Out[49]:

```
['ride fool hostel',
 'cumberland great kitchen',
 'common area comfi bed',
 'nice folk',
 'nearbi mountain bike',
 'big mountain']
```

In [50]:

```
# Revise the previous dataframe transform function.
def newDataFrameTransformation(hotelDf, reviewDF, k=50):
    reviews = reviewDF['reviewCol'].values

    # Top-k frequent terms
    counter = Counter()
    for review in reviews:
            counter.update(flatten([word
                            for word
                            in get_terms(chunker.parse(pos_tag(re.findall(r'\w+', review))))
                            ]))
    topk = counter.most_common(k)

    #Find out if a particular review has the word from topk list
    freqReview = []
    for i in range(len(reviews)):
        tempCounter = Counter(flatten([word
                                       for word
                                       in get_terms(chunker.parse(pos_tag(re.findall(r'\w+',reviews
i])))))]))
        topkinReview = [1 if tempCounter[word] > 0 else 0 for (word,wordCount) in topk]
        freqReview.append(topkinReview)


    #Prepare freqReviewDf
    freqReviewDf = pd.DataFrame(freqReview)
    dfName = []
    for c in topk:
        dfName.append(c[0])
    freqReviewDf.columns = dfName
    finalreviewDf = reviewDF.join(freqReviewDf)
    #print(topk)
    finaldf = hotelDf[['Hotel_Name','Ratings','Experience']].merge(finalreviewDf)

    return topk, finaldf
```

In [51]:

```
reviewNegDF = reviewDF[reviewDF['Experience']=='negative']
#reviewNegDF
```

In [52]:

```
reviewPosDF = reviewDF[reviewDF['Experience']=='positive']
#reviewPosDF
```

In [53]:

```
topk_pos_phrase, finaldf_pos_phrase = newDataFrameTransformation(reviews_df, reviewPosDF)
```

In [54]:

```
topk_neg_phrase, finaldf_neg_phrase = newDataFrameTransformation(reviews_df, reviewNegDF)
```

In [55]:

```
topk_phrase, finaldf_phrase = newDataFrameTransformation(reviews_df, reviewDF)
```

In [57]:

```
for item_a, item_b in zip(topk_pos_phrase, topk_neg_phrase):
    print(item_a,"                    ",item_b)
```

```
('port alberni', 224)                    ('front desk', 88)
('hot tub', 153)                       ('port alberni', 36)
('great place', 127)                       ('hot tub', 22)
('front desk', 116)                      ('credit card', 21)
('bedroom suit', 98)                       ('next day', 21)
('vancouv island', 94)                       ('park lot', 18)
('beach club resort', 86)                       ('next morn', 16)
('beach club', 84)                      ('nthe room', 16)
('ocean view', 73)                      ('night stay', 16)
('friendli staff', 65)                       ('hot water', 16)
('full kitchen', 62)                      ('continent breakfast', 15)
('front desk staff', 56)                       ('front desk staff', 13)
('live room', 52)                     ('coffe maker', 12)
('first time', 50)                     ('first time', 12)
('comfort bed', 48)                      ('good thing', 11)
('great locat', 47)                      ('first room', 11)
('great time', 47)                     ('queen bed', 11)
('next year', 47)                     ('beach club resort', 11)
('short walk', 45)                     ('hotel room', 10)
('great view', 45)                     ('first night', 10)
('queen bed', 44)                     ('live room', 10)
('great stay', 43)                     ('beach club', 10)
('hospit inn', 43)                     ('next time', 9)
('indoor pool', 40)                      ('doubl bed', 9)
('mini golf', 40)                     ('air condition', 9)
('good valu', 39)                     ('custom servic', 9)
('nthe room', 38)                     ('free wifi', 8)
('next time', 36)                     ('good locat', 8)
('night stay', 36)                     ('air condit', 8)
('nice place', 36)                     ('nice view', 8)
('short drive', 35)                      ('ice machin', 8)
('beach acr', 35)                     ('room servic', 8)
('minut drive', 33)                      ('ground floor', 8)
('best western', 33)                       ('main build', 8)
('room servic', 33)                      ('best western', 8)
('help staff', 32)                     ('poco inn', 8)
('easi access', 31)                      ('nthe staff', 7)
('rathtrevor beach', 31)                          ('differ room', 7)
('hotel staff', 29)                      ('bedroom suit', 7)
('oceansid villag resort', 29)                          ('good valu', 7)
('great experi', 28)                       ('clean staff', 7)
('next morn', 28)                     ('main road', 7)
('eagl nook', 28)                     ('vancouv island', 7)
('good food', 27)                     ('hospit inn', 7)
('wonder stay', 27)                      ('nice place', 7)
('coffe maker', 27)                      ('hair dryer', 7)
('hotel room', 27)                     ('hotel restaur', 6)
('perfect place', 26)                       ('friendli staff', 6)
('second floor', 25)                      ('long time', 6)
('first night', 25)                      ('whole place', 6)
```

Repeating this analysis for the top-50 noun phrases and summarization of findings:

The positive reviews contained more objectively positive phrases that were used such as "great time", "great view", infact even though the word "great" appeared in the top negative reviews, the phrases only appears in the positive context. The phrases in the negative list are sometimes positive phrases (like, "good thing", "good location", "best western", "clean staff")- this goes to show that people generally tend to say positive things about a hotel while saying negative things about it. The phrases made it easier to figure out the specific aspects of the hotel industries that people mostly talk about/interact with, like the "front desk" and they often talk about how their stay affects the "next time".

Mutual Information:

(a) Use mutual information (MI) with ground truth sentiment to rank the top-50 most sentiment- bearing non-stopwords in the review collection.

In [58]:

```
def dataFrameTransformation(hotelDf, reviewDF, k=50):
    reviews = reviewDF['reviewCol'].values

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

    # Top-k frequent terms
    counter = Counter()
    for review in reviews:
            counter.update([word.lower()
                            for word
                            in re.findall(r'\w+', review)
                            if word.lower() not in stop and len(word) > 2])
    topk = counter.most_common(k)

    #Find out if a particular review has the word from topk list
    freqReview = []
    for i in range(len(reviews)):
        tempCounter = Counter([word.lower() for word in re.findall(r'\w+',reviews[i])])
        topkinReview = [1 if tempCounter[word] > 0 else 0 for (word,wordCount) in topk]
        freqReview.append(topkinReview)

    #Prepare freqReviewDf
    freqReviewDf = pd.DataFrame(freqReview)
    dfName = []
    for c in topk:
        dfName.append(c[0])
    freqReviewDf.columns = dfName
    finalreviewDf = reviewDF.join(freqReviewDf)
    finaldf = hotelDf[['Hotel_Name','Ratings','Experience']].merge(finalreviewDf)
    return topk, finaldf
```

In [59]:

```
topk, finaldf = dataFrameTransformation(reviews_df, reviewDF, k=50)
```

In [63]:

```
topk[:10]
```

Out[63]:

```
[('room', 3445),
 ('stay', 2198),
 ('great', 2144),
 ('hotel', 2143),
 ('clean', 1874),
 ('staff', 1788),
 ('would', 1540),
 ('good', 1440),
 ('beach', 1407),
 ('stayed', 1398)]
```

In [64]:

```
tempDF_pos = reviews_df[reviews_df['Experience']=='positive']
temp_topk, temp_finaldf = dataFrameTransformation(tempDF_pos, reviewDF, k=50)
```

In [65]:

```
def getMI(topk, df, label_column='Experience'):
    miScore = []
    for word in topk:
        miScore.append([word[0]]+[metrics.mutual_info_score(df[label_column], df[word[0]])])
    miScoredf = pd.DataFrame(miScore).sort_values(1,ascending=0)
    miScoredf.columns = ['Word','MI Score']
    return miScoredf
```

In [66]:

```
#MI_neg = getMI(result_neg, finaldf)
#MI_pos = getMI(result_pos, finaldf)
MI_topk = getMI(topk,finaldf)
```

In [67]:

```
MI_topk.head()
```

Out[67]:

	Word	MI Score
2	great	0.004182
8	beach	0.003794
0	room	0.002913
3	hotel	0.002242
19	night	0.002160

In [68]:

```
#temp_topk, temp_finaldf
MI_topk = getMI(temp_topk, temp_finaldf)
```

In [70]:

```
MI_topk.head()
```

Out[70]:

	Word	MI Score
43	recommend	1.804112e-15
25	service	1.776357e-15
16	comfortable	1.776357e-15
34	two	1.776357e-15
32	get	1.776357e-15

Interesting and/or locale-specific aspects of these top-ranked words
The scores for mutual information are low which goes to say that there is small dependency between the word and the ground truth of the review. Another aspect of the list is that it was taken from the entire collection of reviews without filtering the positive and negative reviews. This is especially problematic because some of the words (like great) appear in both positive and negative reviews a number of times, hence confusing the model.
It should also be noted that the topk words retrieved from the previous function is repeated here and this is due the fact that the MI takes into consideration the number of times a word appreas in the given collection. Therefore, the more frequent a word appears, the more information it would seem to have.

(b) Repeating this analysis for the top-50 noun phrases

In [71]:

```
MI_phrases = getMI(topk_phrase, finaldf_phrase)
```

In [72]:

```
MI_phrases.head()
```

Out[72]:

	Word	MI Score
1	front desk	0.001611
4	bedroom suit	0.001110
3	great place	0.000901
11	full kitchen	0.000692
8	ocean view	0.000662

The Mutual information scores are significantly smaller than MI scores of induvidual words. This goes to say that the phrases present here are indpendent of the information (positive or negative) they are trying to convey. Furthermore, the number of times the phrases occur in the collection is significantly smaller than the induvidual words occuring. This also affects the MI score. The phrases picked out, like "mini golf" or "indoor pool" are relevant to only a few hotels and its reviews, and therefore cannot be an indicator of how positive or negative the entire hotel staying experience would be.

Pointwise Mutual Information

PMI for the top-50 words with positive and negative reviews.

In [73]:

```
def pmiCal(df, x):
    pmilist=[]
    for i in (['positive','negative']):
        for j in [0,1]:
            px = sum(df['Experience']==i)/len(df)
            py = sum(df[x]==j)/len(df)
            pxy = len(df[(df['Experience']==i) & (df[x]==j)])/len(df)
            if pxy==0:#Log 0 cannot happen
                pmi = math.log((pxy+0.0001)/(px*py))
            else:
                pmi = math.log(pxy/(px*py))
            pmilist.append([i]+[j]+[px]+[py]+[pxy]+[pmi])
    pmidf = pd.DataFrame(pmilist)
    pmidf.columns = ['x','y','px','py','pxy','pmi']
    return pmidf
```

In [74]:

```
pmiCal(finaldf,'hotel')
```

Out[74]:

	x	y	px	py	pxy	pmi
0	positive	0	0.969886	0.664623	0.650177	0.008601
1	positive	1	0.969886	0.335377	0.319709	-0.017267
2	negative	0	0.030114	0.664623	0.014446	-0.326035
3	negative	1	0.030114	0.335377	0.015668	0.439129

In [75]:

```
def pmiIndivCal(df,x,gt, label_column='Experience'):
    px = sum(df[label_column]==gt)/len(df)
    py = sum(df[x]==1)/len(df)
    pxy = len(df[(df[label_column]==gt) & (df[x]==1)])/len(df)
    if pxy==0:#Log 0 cannot happen
        pmi = math.log((pxy+0.0001)/(px*py))
    else:
        pmi = math.log(pxy/(px*py))
    return pmi
```

In [76]:

```
pmiIndivCal(finaldf,'hotel','positive')
```

Out[76]:

```
-0.017267140740509255
```

In [77]:

```
def pmiForAllCal(df, label_column='Experience', topk=topk):
    #Try calculate all the pmi for top k and store them into one pmidf dataframe
    pmilist = []
    pmiposlist = []
    pmineglist = []
    for word in topk:
        pmilist.append([word[0]]+[pmiCal(df,word[0])])
        pmiposlist.append([word[0]]+[pmiIndivCal(df,word[0],'positive',label_column)])
        pmineglist.append([word[0]]+[pmiIndivCal(df,word[0],'negative',label_column)])
    pmidf = pd.DataFrame(pmilist)
    pmiposlist = pd.DataFrame(pmiposlist)
    pmineglist = pd.DataFrame(pmineglist)
    pmiposlist.columns = ['word','pmi']
    pmineglist.columns = ['word','pmi']
    pmidf.columns = ['word','pmi']
    return pmiposlist, pmineglist, pmidf
```

In [78]:

```
pmiposlist, pmineglist, pmidf = pmiForAllCal(finaldf)
```

In [79]:

```
#Sorted top pmi words for positive reviews
pmiposlist.sort_values('pmi',ascending=0).head()
```

Out[79]:

	word	pmi
48	beautiful	0.020880
31	helpful	0.020150
8	beach	0.017515
2	great	0.017327
16	comfortable	0.015198

In [82]:

```
#Sorted top pmi words for positive reviews
pmineglist.sort_values('pmi',ascending=0).head()
```

Out[82]:

	word	pmi
29	quot	0.869468
19	night	0.666707
32	get	0.626517
17	breakfast	0.583484
39	could	0.517864

PMI concluding remarks:

The PMI scores are higher than just the mutual information scores, this is because mutual information scores are an average of all the senarios in which the term and the emotion are present and as previously established, there is considerable over lap of words being present in both the lists. PMI however involves itself for the word in the positive or negative context only, the ambiguity is removed. Therefore, the number of words present from a postive or negative review

is already classified and word is then better associated with that classification.

In [83]:

```
pmiposlist_phrase, pmineglist_phrase, pmidf_phrase = pmiForAllCal(finaldf_phrase,topk =
topk_phrase)
```

In [84]:

```
pmiposlist_phrase.sort_values('pmi',ascending=0).head()
```

Out[84]:

	word	pmi
47	easi access	0.030577
11	full kitchen	0.028695
3	great place	0.028677
44	help staff	0.027929
45	rathtrevor beach	0.026653

In [85]:

```
pmineglist_phrase.sort_values('pmi',ascending=0).head()
```

Out[85]:

	word	pmi
40	continent breakfast	1.762871
27	next morn	1.290543
32	next day	1.270637
30	park lot	1.175454
1	front desk	1.041710

PMI for top-50 noun phrases concluding notes:

The same phrases were repeated for the phrases for the positive and negative reviews, with opposite signs. This is because the top K phrases are gotten from the same dataframe, and are mapped to opposite classes (positive and negative), The topk words remain constant as they are from the same dataframe, but the context they are applied to are different (positive/negative) - hence the opposite signs for the different lists.
The values are different because of the number of times a certain phrase is repeated in the positive context is different then the number of times it is repeated in the negative context.
Over all the values for the PMI was larger than that for just MI because mututal information was an average of all the occurrence of the terms and a lot of the times, the terms exisited in both the positive and the negative context, hence the words were not able to learn if its occurrence was to do with positive or the negative class. For PMI, we are splitting it into the negative and positive lists, therefore if a phrase exists in a negative tweet, it has more information to give with regards to the negative tweet.

Analysis repeated for the single top and single bottom hotel (according to the ground truth rating).

In [86]:

```
# Single top hotel: Stansbury&#39;s Guest House
# Single bottom hotel: Howard Johnson Hotel Port Alberni
```

In [87]:

```
best_hotel_df = reviews_df[reviews_df['Hotel_Name']=='Stansbury&#39;s Guest House']
worst_hotel_df = reviews_df[reviews_df['Hotel_Name']=='Howard Johnson Hotel Port Alberni']
best_hotel_df.head()
```

Out[87]:

	Link	Hotel_Name	Review text	Ratings	Experience
28	data/ca/1015432/3589554/163624982.html	Stansbury's Guest House	"Stayed for a week in the Stansbury\\'s guest ...	5	positive
29	data/ca/1015432/3589554/165371253.html	Stansbury's Guest House	"A group of 6 of us stayed here for a weekend ...	5	positive
30	data/ca/1015432/3589554/169057536.html	Stansbury's Guest House	"Visiting from New Zealand,we were fortunate t...	5	positive
31	data/ca/1015432/3589554/210674359.html	Stansbury's Guest House	"Gwen and Scott have a beautiful Guest House a...	5	positive
32	data/ca/1015432/3589554/215774198.html	Stansbury's Guest House	"A beautifully finished apartment with every c...	5	positive

In [88]:

```
BestWorst_DF = best_hotel_df.append(worst_hotel_df)
BestWorst_DF
BestWorst_DF.rename(columns={'Review text':'reviewCol'},inplace=True)
BestWorst_DF.head()
```

Out[88]:

	Link	Hotel_Name	reviewCol	Ratings	Experience
28	data/ca/1015432/3589554/163624982.html	Stansbury's Guest House	"Stayed for a week in the Stansbury\\'s guest ...	5	positive
29	data/ca/1015432/3589554/165371253.html	Stansbury's Guest House	"A group of 6 of us stayed here for a weekend ...	5	positive
30	data/ca/1015432/3589554/169057536.html	Stansbury's Guest House	"Visiting from New Zealand,we were fortunate t...	5	positive
31	data/ca/1015432/3589554/210674359.html	Stansbury's Guest House	"Gwen and Scott have a beautiful Guest House a...	5	positive
32	data/ca/1015432/3589554/215774198.html	Stansbury's Guest House	"A beautifully finished apartment with every c...	5	positive

In [89]:

```
Ratings = BestWorst_DF['Ratings']
reviewDF = pd.concat([reviewDF, Ratings],axis =1 )
#reviewDF.rename(columns={'exp_index':'Ratings'},inplace=True)
```

In [90]:

```
topk_bestworst, final_bestworstDF = dataFrameTransformation(reviewDF,BestWorst_DF, k=50)
topk_bestworst[:10]
```

Out[90]:

```
[('room', 110),
 ('stay', 80),
 ('hotel', 64),
 ('clean', 61),
 ('gwen', 60),
 ('great', 59),
 ('place', 55),
 ('scott', 46),
 ('stayed', 45),
 ('cumberland', 43)]
```

In [91]:

```
def pmiCall(df, x):
    pmilist=[]
    for i in ['positive','negative']:
        for j in [0,1]:
            pr = sum(df['Experience']==i)/len(df)
```

```
            px = sum(df['Experience']==i)/len(df)
            py = sum(df[x]==j)/len(df)
            pxy = len(df[(df['Experience']==i) & (df[x]==j)])/len(df)
            if pxy==0:#Log 0 cannot happen
                if (px == 0 or py == 0):
                    pmi = math.log((pxy+0.0001)/(px*py+0.0001))
                else:
                    pmi = math.log((pxy+0.0001)/(px*py))
            else:
                pmi = math.log(pxy/(px*py))
            pmilist.append([i]+[j]+[px]+[py]+[pxy]+[pmi])
    pmidf = pd.DataFrame(pmilist)
    pmidf.columns = ['x','y','px','py','pxy','pmi']
    return pmidf
```

In [92]:

```
def pmiIndivCal1(df,x,gt, label_column='Experience'):
    pmi = 0
    px = sum(df[label_column]==gt)/len(df)
    py = sum(df[x]==1)/len(df)
    pxy = len(df[(df[label_column]==gt) & (df[x]==1)])/len(df)
    if pxy==0:#Log 0 cannot happen
        if (px ==0 or py == 0):
            pmi = math.log((pxy+0.0001)/(px*py+0.0001))
        else:
            pmi = math.log((pxy+0.0001)/(px*py))
    else:
        pmi = math.log(pxy/(px*py))
    return pmi
```

In [93]:

```
def pmiForAllCal1(df, label_column='Experience', topk=topk):
    #Try calculate all the pmi for top k and store them into one pmidf dataframe
    pmilist = []
    pmiposlist = []
    pmineglist = []
    for word in topk:
        pmilist.append([word[0]]+[pmiCal1(df,word[0])])
        pmiposlist.append([word[0]]+[pmiIndivCal1(df,word[0],'positive',label_column)])
        pmineglist.append([word[0]]+[pmiIndivCal1(df,word[0],'negative',label_column)])
    pmidf = pd.DataFrame(pmilist)
    pmiposlist = pd.DataFrame(pmiposlist)
    pmineglist = pd.DataFrame(pmineglist)
    pmiposlist.columns = ['word','pmi']
    pmineglist.columns = ['word','pmi']
    pmidf.columns = ['word','pmi']
    return pmiposlist, pmineglist, pmidf
```

In [94]:

```
pmiposlist_bestHotel, pmineglist_bestHotel, pmidf_bestHotel = pmiForAllCal(final_bestworstDF, topk
= topk_bestworst)
```

In [95]:

```
#Sorted top pmi words for positive reviews
y = pmiposlist_bestHotel.sort_values('pmi',ascending=0)
y
```

Out[95]:

	word	pmi
0	room	0.351376
37	nice	0.351376
27	back	0.351376
28	hosts	0.351376
29	location	0.351376

	word	pmi
30	could	0.351376
31	well	0.351376
32	bathroom	0.351376
33	home	0.351376
34	town	0.351376
35	much	0.351376
36	old	0.351376
38	everything	0.351376
1	stay	0.351376
39	also	0.351376
40	even	0.351376
41	perfect	0.351376
42	like	0.351376
43	nthe	0.351376
44	amp	0.351376
45	suite	0.351376
46	night	0.351376
47	get	0.351376
48	trails	0.351376
26	two	0.351376
25	comfortable	0.351376
24	quot	0.351376
13	would	0.351376
2	hotel	0.351376
3	clean	0.351376
4	gwen	0.351376
7	scott	0.351376
8	stayed	0.351376
9	cumberland	0.351376
10	staff	0.351376
11	rooms	0.351376
23	mountain	0.351376
12	house	0.351376
15	guest	0.351376
16	stansbury	0.351376
17	good	0.351376
18	friendly	0.351376
19	bike	0.351376
20	time	0.351376
22	helpful	0.351376
49	area	0.351376
14	breakfast	0.351376
6	place	0.351376
5	great	0.351376
21	one	0.351376

In [96]:

```
#Sorted top pmi words for positive reviews
x = pmineglist bestHotel.sort values('pmi',ascending=0)
```

```
x.head()
```

Out[96]:

	word	pmi
24	quot	-3.544844
44	amp	-4.679174
43	nthe	-4.713661
32	bathroom	-4.931138
46	night	-4.931138

In [97]:

```
topk_best1, final_bestDF1 = dataFrameTransformation(best_hotel_df,reviewDF, k=50)
#topk_best1
```

In [98]:

```
pmiposlist_bestHotel, pmineglist_bestHotel, pmidf_bestHotel = pmiForAllCal1(final_bestDF1, topk = t
opk_best1)
```

Hotel-specic insights about what is good and bad about these two hotels:

The positive list did contain valuable information such that the guests particularly like their stay depending on their "breakfast" or the "place", or the "location. The positive PMI value indicates that there was a strong correlation between the words and ratings they got. The negative list however, proved to be less informative as all the values were negative, which indicates that the terms cooccured less frequently with the negative class. This could be due to the fact that there were perhaps not many entries in the database to derive enough assupmtions.

Q5. General Plots

(a) Histogram of ground truth and vader sentiments scores:

In [100]:

```
x = reviews_df['Ratings']
x_df = pd.DataFrame(x)
review_temp_df = reviewDF.copy()
reviewDF_graphs = pd.concat([review_temp_df, x_df], axis = 1)
reviewDF_graphs.head()
cols = []
count = 1
for column in reviewDF_graphs.columns:
    if column == 'Ratings':
        cols.append('Ratings_'+str(count))
        count+=1
        continue
    cols.append(column)
reviewDF_graphs.columns = cols
reviewDF_graphs.head()
```

Out[100]:

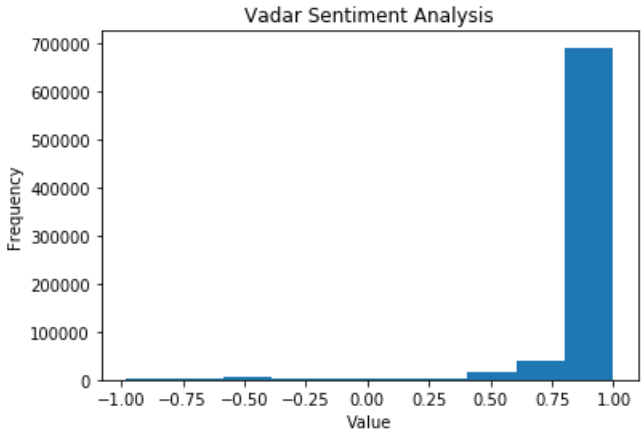
	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings_1	Ratings_2
0	"I was looking for a place to sit and chill fo...	0.8442	positive	The Riding Fool Hostel	1	NaN	5
1	"I stayed at the Riding Fool Hostel whilst I w...	0.9965	positive	The Riding Fool Hostel	1	NaN	5
2	"My husband and I (both in our 50\\'s) stayed ...	0.9757	positive	The Riding Fool Hostel	1	NaN	5
3	"We were warmly welcomed by Caitlin, the new ...	0.9766	positive	The Riding Fool Hostel	1	NaN	5
4	"Comfortable, cheap, and cosy accommodation wi...	0.4939	positive	The Riding Fool Hostel	1	NaN	5

In [101]:

```
def getHistogram(measure, title):
    if measure=='both':
        x = [finaldf['Ratings'].values/5]
        y = [finaldf['vader'].values]
        bins = np.linspace(-1, 1, 100)
        plt.title(title)
        plt.hist(x, bins, label='Ratings')
        plt.hist(y, bins, label='Vader Score')
        plt.legend(loc='upper right')
        plt.show()

    else:
        plt.hist(finaldf[measure].values)
        plt.title(title)
        plt.xlabel("Value")
        plt.ylabel("Frequency")
        fig = plt.gcf()
```

In [102]:

```
getHistogram('vader', 'Vadar Sentiment Analysis')
```



In [103]:

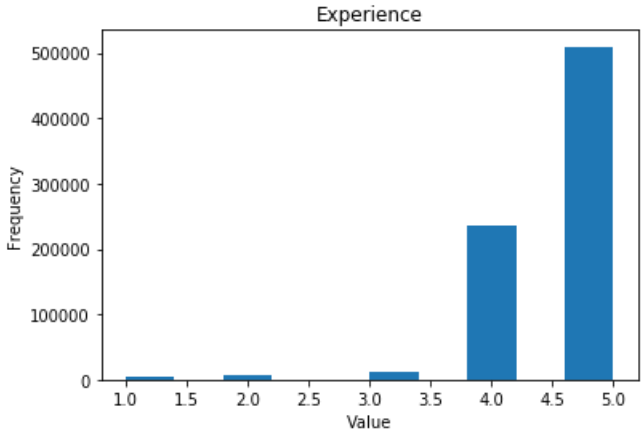
```
x = finaldf['Ratings'].values
x = finaldf['Hotel_Name'].values
type(x[5])
```

Out[103]:

```
str
```

In [104]:

```
getHistogram('Ratings', 'Experience')
```

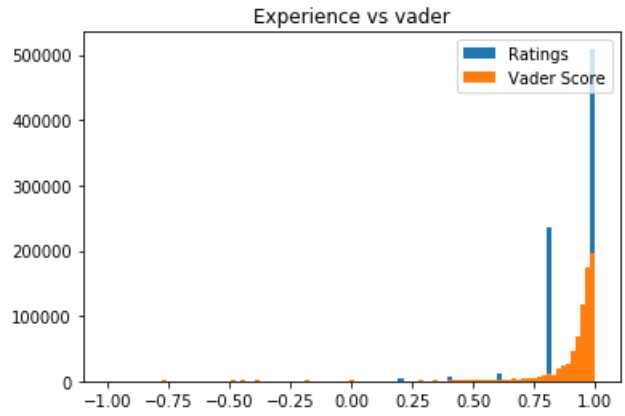


Differences

While people experience (1-5 ratings) was quantified, the actual remarks given by the people paint a different story. This is in line with a previous observation, that even when giving a hotel a negative remark, there is some aspect of the review which is a positive remark followed by a "but". Therefore, it is fathomable that hotels with lower ratings would also be given a higher vader score, if their review had some positive comment in it.

In [105]:

```
getHistogram('both', 'Experience vs vader')
```



In [106]:

```
!pip install plotly
```

```
Requirement already satisfied: plotly in /Users/krutheekarajkumar/anaconda3/lib/python3.7/site-
packages (4.1.1)
Requirement already satisfied: retrying>=1.3.3 in
/Users/krutheekarajkumar/anaconda3/lib/python3.7/site-packages (from plotly) (1.3.3)
Requirement already satisfied: six in /Users/krutheekarajkumar/anaconda3/lib/python3.7/site-
packages (from plotly) (1.12.0)
```

In [107]:

```
import plotly.graph_objs as go
```

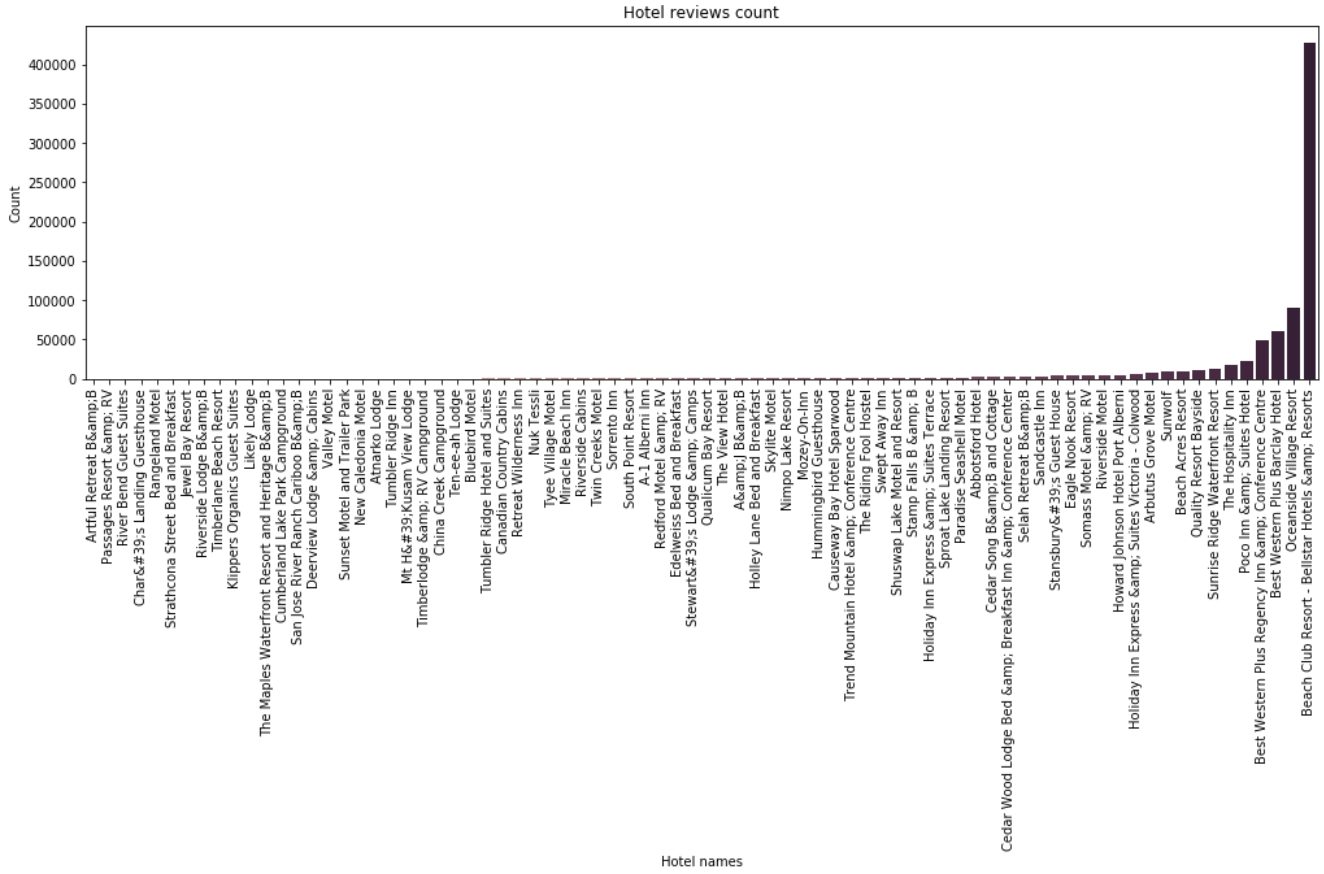
In [108]:

```
a = finaldf['Hotel_Name']

#df.sort_values(by='val', ascending=False)

def count_elements(seq) -> dict:
    """Tally elements from `seq`."""
    hist = {}
    for i in seq:
        hist[i] = hist.get(i, 0) + 1
    return hist

count = count_elements(a)
keys = list(count.keys())
vals = list(count.values())
#count_sorted = sorted(count.values())
type(keys)
data_tuples = list(zip(keys,vals))
count = pd.DataFrame(data_tuples, columns=['Hotel','count'])
count_sorted = count.sort_values(by = ['count'])
```

In [109]:

```
fig, ax = plt.subplots(figsize=(17,5))
plt.setp(plt.xticks()[1], rotation=90)
sns.catplot(x='Hotel', y ='count', palette="ch:.25", kind= 'bar', data=count_sorted, ax = ax);
```

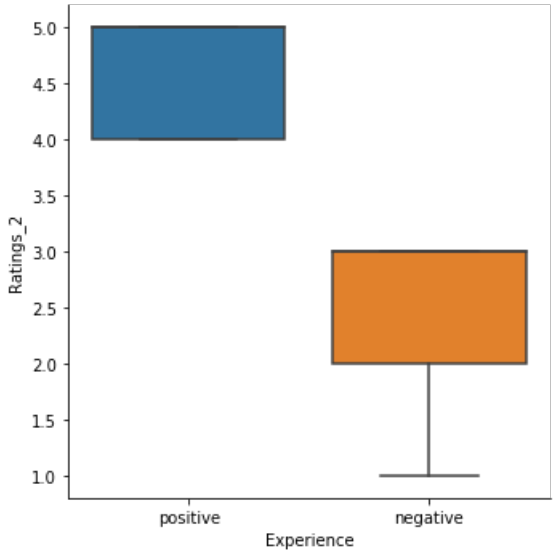
```
plt.close(2)
plt.title('Hotel reviews count')
ax.set(xlabel='Hotel names', ylabel='Count')
plt.show()
```



(b) Boxplots for ground truth and vader sentiment:

In [110]:

```
sns.catplot(x="Experience", y="Ratings_2", kind="box", data=reviewDF_graphs);
```



In [111]:

```
reviewDF_graphs
```

Out[111]:

	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings_1	Ratings_2
0	"I was looking for a place to sit and chill fo...	0.8442	positive	The Riding Fool Hostel	1	NaN	5
1	"I stayed at the Riding Fool Hostel whilst I w...	0.9965	positive	The Riding Fool Hostel	1	NaN	5
2	"My husband and I (both in our 50\\'s) stayed ...	0.9757	positive	The Riding Fool Hostel	1	NaN	5
3	"We were warmly welcomed by Caitlin, the new ...	0.9766	positive	The Riding Fool Hostel	1	NaN	5
4	"Comfortable, cheap, and cosy accommodation wi...	0.4939	positive	The Riding Fool Hostel	1	NaN	5
5	"This is a fun place to stay in a friendly lit...	0.9842	positive	The Riding Fool Hostel	1	NaN	5
6	"My wife and I stayed here for 4 nights to rid...	0.9370	positive	The Riding Fool Hostel	1	NaN	5
7	"When compared to other hostels on the west co...	0.9692	positive	The Riding Fool Hostel	1	NaN	5
8	"This place is amazing. Lovely old historic bu...	0.9960	positive	The Riding Fool Hostel	1	NaN	5
9	"I loved my time here. The hostel has a great ...	0.9565	positive	The Riding Fool Hostel	1	NaN	5
10	"Considering this place has a star from the 20...	0.1021	negative	The Riding Fool Hostel	0	NaN	2
11	"Had a great experience at this hostel! The st...	0.9170	positive	The Riding Fool Hostel	1	NaN	5
12	"The common area is awesome. Had a great few d...	0.9392	positive	The Riding Fool Hostel	1	NaN	5
13	"I couldn\\'t agree more with the other positi...	0.9885	positive	The Riding Fool Hostel	1	NaN	5
14	"The hostel itself is good and clean with a hu...	0.3612	negative	The Riding Fool Hostel	0	NaN	3
15	"I should have brought my mountain bike on thi...	0.9694	positive	The Riding Fool Hostel	1	NaN	5
16	"Arrived August 6th 2016 and the first thing I...	0.9891	positive	The Riding Fool Hostel	1	NaN	4
17	"Our two night stay alone in a three bed room ...	0.9787	positive	The Riding Fool Hostel	1	NaN	5
18	"Second time my boyfriend and i stayed here an...	0.9432	positive	The Riding Fool Hostel	1	NaN	5
19	"Super friendly staff. Really clean and an awe...	-0.7205	positive	The Riding Fool Hostel	1	NaN	4
20	"My biggest complaint with the Riding Fool is ...	0.8000	negative	The Riding Fool Hostel	0	NaN	3
21	"We have stayed at Riding Fool twice now. Bot...	0.6028	positive	The Riding Fool Hostel	1	NaN	5
22	"I had my first hosteling experience at the Ri...	0.4915	positive	The Riding Fool Hostel	1	NaN	5
23	"Stayed in Room 7 recently. Noise from the sta...	0.7506	negative	The Riding Fool Hostel	0	NaN	3
24	"Great hostel, very clean, well run, good faci...	0.9704	positive	The Riding Fool Hostel	1	NaN	5
25	"This hostel occupies the first floor (upstair...	0.9942	positive	The Riding Fool Hostel	1	NaN	5
26	"We stayed here as a stop off on our way to No...	0.9022	positive	The Riding Fool Hostel	1	NaN	5
27	"We stayed for one night here as a stop off be...	0.9825	positive	The Riding Fool Hostel	1	NaN	4
28	"Stayed for a week in the Stansbury\\'s guest ...	0.8682	positive	Stansbury's Guest House	1	5.0	5
29	"A group of 6 of us stayed here for a weekend ...	0.9367	positive	Stansbury's Guest House	1	5.0	5
...	...	...	...	...	...	...	...
4029	"My family had a very comfortable stay here. O...	0.9843	positive	Paradise Seashell Motel	1	NaN	5
4030	"I haven\\'t been to this motel but I can cert...	-0.9493	negative	Paradise Seashell Motel	0	NaN	1
4031	"Hotel was pretty basic but the rates were ver...	0.9793	positive	Paradise Seashell Motel	1	NaN	4
4032	"We stay in one bed room it is cheap but reall...	0.1154	negative	Paradise Seashell Motel	0	NaN	1
4033	"Motels have always been something that concer...	0.9780	positive	Paradise Seashell Motel	1	NaN	5
4034	"Paradise Seashell Motel is in an excellent lo...	0.9958	negative	Paradise Seashell Motel	0	NaN	3
4035	"We stayed at this busy place on our way to To...	0.9638	positive	Paradise Seashell Motel	1	NaN	4

	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings_1	Ratings_2
4036	"We stayed September long before school starte...	0.9686	positive	Paradise Seashell Motel	1	NaN	4
4037	"My 11 year old son & I took a trip to Par...	0.9963	positive	Paradise Seashell Motel	1	NaN	5
4038	"We stayed here a couple of weeks ago for two ...	0.9466	positive	Paradise Seashell Motel	1	NaN	5
4039	"The only reason I rated this a 3 was that it ...	0.8834	negative	Paradise Seashell Motel	0	NaN	3
4040	"This simple roadside motel puts the bar very ...	0.9578	positive	Paradise Seashell Motel	1	NaN	5
4041	"We come to Parksville every summer and stay f...	0.9541	positive	Paradise Seashell Motel	1	NaN	4
4042	"My teenagers and I spent a night here on our ...	0.5346	positive	Paradise Seashell Motel	1	NaN	4
4043	"Clean rooms, close to the beach and free mini...	0.8402	negative	Paradise Seashell Motel	0	NaN	3
4044	"This hotel is small but clean enough. In the ...	-0.5712	negative	Paradise Seashell Motel	0	NaN	3
4045	"We stayed one night on our way to Tofino and ...	0.9092	positive	Paradise Seashell Motel	1	NaN	5
4046	"Stayed at this location for 3 nights. The un...	0.9423	positive	Paradise Seashell Motel	1	NaN	4
4047	"We stayed a couple nights just before New Yea...	0.8977	negative	Paradise Seashell Motel	0	NaN	3
4048	"We required an overnight on Christmas Day to ...	0.8680	positive	Paradise Seashell Motel	1	NaN	4
4049	"We have stayed there twice now; both times fo...	0.9161	positive	Paradise Seashell Motel	1	NaN	4
4050	"First, this photo is a misrepresentation. Thi...	0.9279	positive	Paradise Seashell Motel	1	NaN	4
4051	"I really like this little place but they have...	0.4471	positive	Paradise Seashell Motel	1	NaN	4
4052	"Had to spend the night unexpectedly in Parksv...	0.8621	positive	Paradise Seashell Motel	1	NaN	5
4053	"While I did not need a full fledged motel roo...	0.2396	positive	Paradise Seashell Motel	1	NaN	4
4054	"Stayed here this year with Grandma and the li...	0.9771	negative	Paradise Seashell Motel	0	NaN	3
4055	"I reserved a room at this hotel. They called ...	-0.9817	negative	Paradise Seashell Motel	0	NaN	1
4056	"Good location and good value for the price. ...	0.9524	positive	Paradise Seashell Motel	1	NaN	4
4057	"The reception staff were friendly and helpful...	0.1655	negative	Paradise Seashell Motel	0	NaN	3
4058	"We stayed here for one night on a week long t...	0.9363	positive	Paradise Seashell Motel	1	NaN	5

4059 rows × 7 columns

In [112]:

```
top_5_GT
stansbury_GT = reviewDF[reviewDF.Hotel_Name == 'Stansbury&#39;s Guest House']
Cedar_Song_GT = reviewDF[reviewDF.Hotel_Name == 'Cedar Song B&amp;B and Cottage']
Stamp_Falls_GT = reviewDF[reviewDF.Hotel_Name == 'Stamp Falls B &amp; B']
Mozey_On_Inn_GT = reviewDF[reviewDF.Hotel_Name == 'Mozey-On-Inn']
Nimpo_Lake_GT = reviewDF[reviewDF.Hotel_Name == 'Nimpo Lake Resort']

gt_df1 = pd.concat([stansbury_GT,Cedar_Song_GT],axis = 0)
gt_df2 = pd.concat([gt_df1,Stamp_Falls_GT],axis = 0)
gt_df3 = pd.concat([gt_df2,Mozey_On_Inn_GT],axis = 0)
gt_df4 = pd.concat([gt_df3,Nimpo_Lake_GT],axis = 0)
```

In [113]:

```
gt_df4
```

Out[113]:

	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings

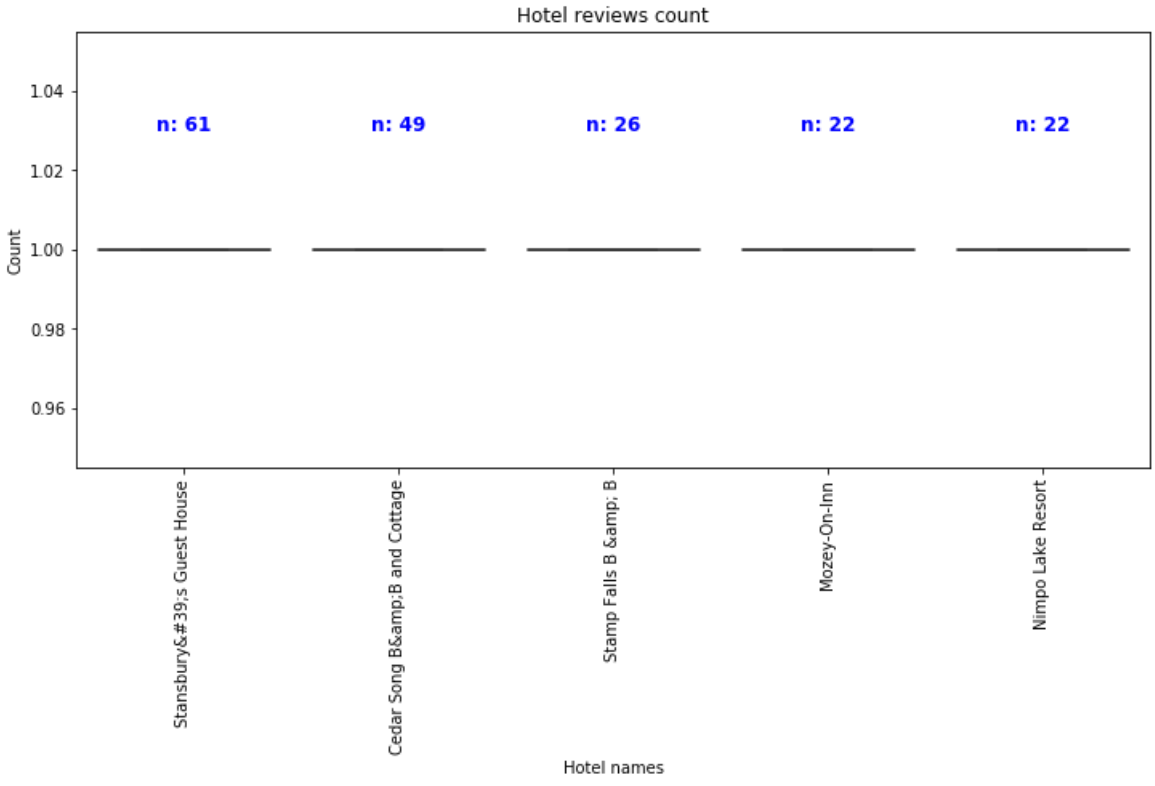
	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings
28	"Stayed for a week in the Stansbury\\'s guest ...	0.8682	positive	Stansbury's Guest House	1	5.0
29	"A group of 6 of us stayed here for a weekend ...	0.9367	positive	Stansbury's Guest House	1	5.0
30	"Visiting from New Zealand,we were fortunate t...	0.9652	positive	Stansbury's Guest House	1	5.0
31	"Gwen and Scott have a beautiful Guest House a...	0.9803	positive	Stansbury's Guest House	1	5.0
32	"A beautifully finished apartment with every c...	0.9807	positive	Stansbury's Guest House	1	5.0
33	"We happended upon Stansbury\\'s last year jus...	0.9860	positive	Stansbury's Guest House	1	5.0
34	"We stayed at Stansbury`s Guest House for 8 da...	0.9940	positive	Stansbury's Guest House	1	5.0
35	"We had an amazing stay at the Stansbury\\'s G...	0.9665	positive	Stansbury's Guest House	1	5.0
36	"Nine of us spent the weekend at the Stansbury...	0.9443	positive	Stansbury's Guest House	1	5.0
37	"This place has: very friendly hosts, is super...	0.9588	positive	Stansbury's Guest House	1	5.0
38	"Cumberland is cute little village in the Como...	0.9638	positive	Stansbury's Guest House	1	5.0
39	"My wife and I stayed at Stansbury\\'s Guest H...	0.8962	positive	Stansbury's Guest House	1	5.0
40	"You will not be disappointed! The rooms and ...	0.8551	positive	Stansbury's Guest House	1	5.0
41	"What a wonderful Guest House to call home for...	0.9273	positive	Stansbury's Guest House	1	4.0
42	"This house was clean and had a beautiful sunn...	0.9716	positive	Stansbury's Guest House	1	5.0
43	"My sister and I just spent 10 days at the Gue...	0.8937	positive	Stansbury's Guest House	1	5.0
44	"I hardly want to share this gem so that Gwen\...	0.9757	positive	Stansbury's Guest House	1	5.0
45	"What a great place! 1, 2, or 3 bedrooms. Bi...	0.9864	positive	Stansbury's Guest House	1	5.0
46	"Right from the beginning Stansbury\\'s Guest ...	0.9532	positive	Stansbury's Guest House	1	5.0
47	"Gwen and Scott.\\n\\nOur stay was Fantastic!!...	0.9576	positive	Stansbury's Guest House	1	5.0
48	"My husband and I stayed at Stansbury\\'s with...	0.9466	positive	Stansbury's Guest House	1	5.0
49	"We have stayed at Stansbury\\'s Guest House a...	0.9744	positive	Stansbury's Guest House	1	5.0
50	"I finally made my way to Cumberland and at th...	0.9836	positive	Stansbury's Guest House	1	5.0
51	"Great location! Close to trails, pubs, food a...	0.9718	positive	Stansbury's Guest House	1	5.0
52	"My Husband and I came for a mountain bike vac...	0.9895	positive	Stansbury's Guest House	1	5.0
53	"Gwen & Scott have created a wonderful san...	0.9897	positive	Stansbury's Guest House	1	5.0
54	"Scott and Gwen made us very welcome. I rode w...	0.8761	positive	Stansbury's Guest House	1	5.0
55	"The hosts are warm and inviting and the 3 bed...	0.9545	positive	Stansbury's Guest House	1	5.0
56	"A perfect place to stay if you want a peacefu...	0.9700	positive	Stansbury's Guest House	1	5.0
57	"Always on the lookout for interesting places ...	0.9881	positive	Stansbury's Guest House	1	5.0
...	...	...	...	...	...	...
731	"Our group was attending a wedding in Granite ...	0.9876	positive	Mozey-On-Inn	1	NaN
732	"A fabulous little gem, the owners passion for...	0.9260	positive	Mozey-On-Inn	1	NaN
733	"What a wonderful, comfortable, and pleasant f...	0.9810	positive	Mozey-On-Inn	1	NaN
734	"Absolute gem of a place. Cleanest room you wi...	0.6808	positive	Mozey-On-Inn	1	NaN
735	"Very charming little gem in Coalmont. We stay...	0.9757	positive	Mozey-On-Inn	1	NaN
736	"We had a great stay in this quiet little town...	0.9537	positive	Mozey-On-Inn	1	NaN
737	"Coalmont is a bit off the beaten path, but th...	0.9908	positive	Mozey-On-Inn	1	NaN
738	"Awesome \\ud83d\\ude0e Wonderful Old Gold Min...	0.9679	positive	Mozey-On-Inn	1	NaN
673	"We stayed in the Pioneer cabin for a week, wh...	0.9839	positive	Nimpo Lake Resort	1	NaN
674	"We stayed at Nimpo Lake Resort from July 6-9,...	0.9785	positive	Nimpo Lake Resort	1	NaN
675	"We only stayed two nights at Nimpo, the rest ...	0.9748	positive	Nimpo Lake Resort	1	NaN
676	"We spent one night in the aptly-named Birdwat...	0.9838	positive	Nimpo Lake Resort	1	NaN
677	"I revisited Nimpo Lake Resort after 14 years,...	0.7717	positive	Nimpo Lake Resort	1	NaN
678	"After a great time last year at Nimpo Lake Re...	0.9794	positive	Nimpo Lake Resort	1	NaN
679	"We booked the Nimpo Lake Resort online and we...	0.9229	positive	Nimpo Lake Resort	1	NaN
680	"We stayed at Nimpo Lake Resort in an RV site;...	0.9909	positive	Nimpo Lake Resort	1	NaN
681	"It\\'s not a hotel!\\nWe had a great time wit...	0.9607	positive	Nimpo Lake Resort	1	NaN
682	"We arrive on the 21st of June 2015. The owne...	0.9493	positive	Nimpo Lake Resort	1	NaN

	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings
683	"Stunning, pristine lake! Fabulous fishing!...	0.9981	positive	Nimpo Lake Resort	1	NaN
684	"Stayed here five nights. The hospitality of ...	0.9500	positive	Nimpo Lake Resort	1	NaN
685	"we stayed two nights at Nimpo Lake Resort. I...	0.7574	positive	Nimpo Lake Resort	1	NaN
686	"Stayed in the pioneer cabin. Right on the la...	0.8268	positive	Nimpo Lake Resort	1	NaN
687	"The cabins contained all the comforts you nee...	0.9060	positive	Nimpo Lake Resort	1	NaN
688	"We stayed in the Birdwatcher\\'s Cabin for fi...	0.9929	positive	Nimpo Lake Resort	1	NaN
689	"My husband and I stayed at the Hilltop Cabin ...	0.9665	positive	Nimpo Lake Resort	1	NaN
690	"We had a wonderful stay at Nimpo Lake Resort....	0.9194	positive	Nimpo Lake Resort	1	NaN
691	"Completely fulfilled expectations! The cabin ...	0.9934	positive	Nimpo Lake Resort	1	NaN
692	"We we\\'re so lucky we drove up with out a re...	0.9394	positive	Nimpo Lake Resort	1	NaN
693	"Had the pleasure of staying at this resort a ...	0.9600	positive	Nimpo Lake Resort	1	NaN
694	"We recently enjoyed a few days at Nimpo Lake ...	0.9798	positive	Nimpo Lake Resort	1	NaN

180 rows × 6 columns

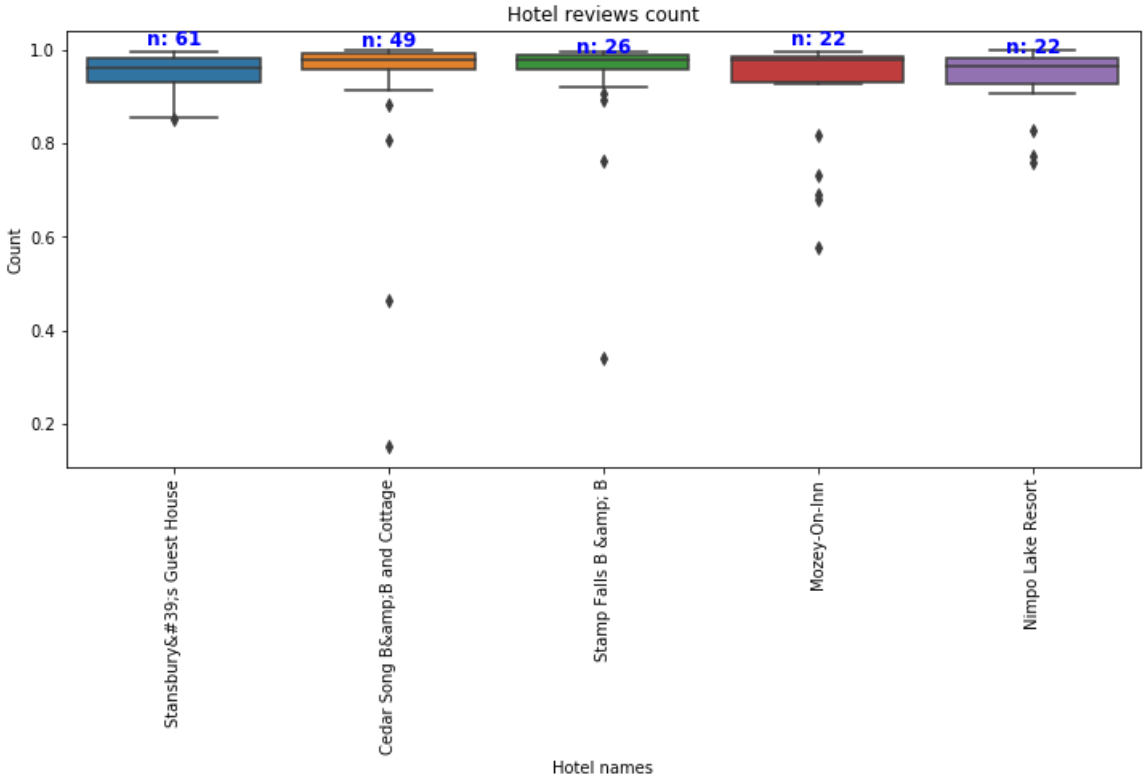
In [112]:

```
fig, ax = plt.subplots(figsize=(12,5))
plt.setp(plt.xticks()[1], rotation=90)
sns.catplot(x="Hotel_Name", y="exp_index", kind="box", data=gt_df4, ax = ax);
plt.close(2)
plt.title('Hotel reviews count')
ax.set(xlabel='Hotel names', ylabel='Count')

# Calculate number of obs per group & median to position labels
medians = gt_df4.groupby(['Hotel_Name'])['exp_index'].median().values
nobs = gt_df4['Hotel_Name'].value_counts().values
nobs = [str(x) for x in nobs.tolist()]
nobs = ["n: " + i for i in nobs]

# Add it to the plot
pos = range(len(nobs))
for tick,label in zip(pos,ax.get_xticklabels()):
    ax.text(pos[tick], medians[tick] + 0.03, nobs[tick],
horizontalalignment='center', size='large', color='b', weight='semibold')

plt.show()
```



In [113]:

```
fig, ax = plt.subplots(figsize=(12,5))
plt.setp(plt.xticks()[1], rotation=90)
sns.catplot(x="Hotel_Name", y="vader", kind="box", data=gt_df4, ax = ax);
plt.close(2)
plt.title('Hotel reviews count')
ax.set(xlabel='Hotel names', ylabel='Count')

# Calculate number of obs per group & median to position labels
medians = gt_df4.groupby(['Hotel_Name'])['vader'].median().values
nobs = gt_df4['Hotel_Name'].value_counts().values
nobs = [str(x) for x in nobs.tolist()]
nobs = ["n: " + i for i in nobs]

# Add it to the plot
pos = range(len(nobs))
for tick,label in zip(pos,ax.get_xticklabels()):
    ax.text(pos[tick], medians[tick] + 0.03, nobs[tick],
horizontalalignment='center', size='large', color='b', weight='semibold')



plt.show()
```



(b) Mean and variance of the ground truth and Vader sentiment scores for the top-5 ranked hotels according to star rating

The mean and the variance for the ground truth was the exact same score - one as all reviews they had gotten were positive. The boxplots for the vader scores were more informative. The variance score is more informative than the mean of the top five hotels. For example, Stamp Falls and Mozey-On-Inn are both highly rated hotels, however Stamp Falls has a higher number of people rated on it and most people had more positive comments to give than in Mozzey-On_Inn.

Variance:

Vader: 0.010745
Ratings: 0.0

Mean:

Vader: 0.94

Ratings:1

(c) Scatterplots and heatmaps of ground truth score (star ratings) versus vader sentiment score.

In [114]:

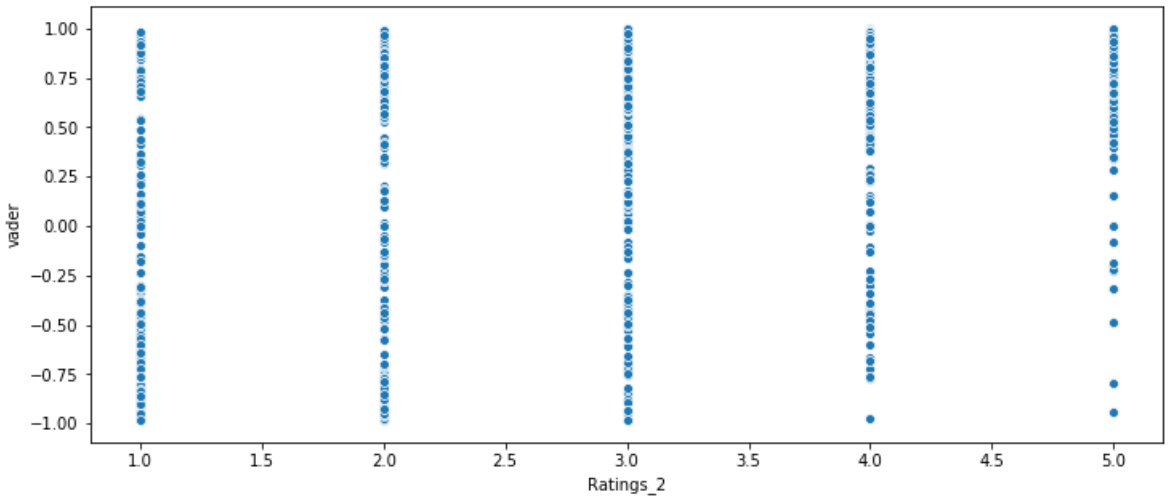
```
temp = reviews_df['Ratings']
scatterDF = pd.concat([reviewDF,temp],axis = 1)
scatterDF.head()
cols = []
count = 1
for column in scatterDF.columns:
    if column == 'Ratings':
        cols.append('Ratings_'+str(count))
        count+=1
        continue
    cols.append(column)
scatterDF.columns = cols
scatterDF.head()
```

Out[114]:

	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings_1	Ratings_2
0	"I was looking for a place to sit and chill fo...	0.8442	positive	The Riding Fool Hostel	1	NaN	5
1	"I stayed at the Riding Fool Hostel whilst I w...	0.9965	positive	The Riding Fool Hostel	1	NaN	5
2	"My husband and I (both in our 50\\'s) stayed ...	0.9757	positive	The Riding Fool Hostel	1	NaN	5
3	"We were warmly welcomed by Caitlin, the new ...	0.9766	positive	The Riding Fool Hostel	1	NaN	5
4	"Comfortable, cheap, and cosy accommodation wi...	0.4939	positive	The Riding Fool Hostel	1	NaN	5

In [115]:

```
fig, ax = plt.subplots(figsize=(12,5))
ax = sns.scatterplot(x="Ratings_2", y="vader", data=scatterDF)
```



In [116]:

```
R = scatterDF['Ratings_2'].values
V = scatterDF['vader'].values
#x, y = np.meshgrid(R, V)
from scipy.stats.kde import gaussian_kde
V
```

Out[116]:

```
array([0.8442, 0.9965, 0.9757, ..., 0.9524, 0.1655, 0.9363])
```

In [117]:

```
#ax3 = sns.heatmap(np.log10(zi.reshape(xi.shape)),cmap=cmap)
```

In [118]:

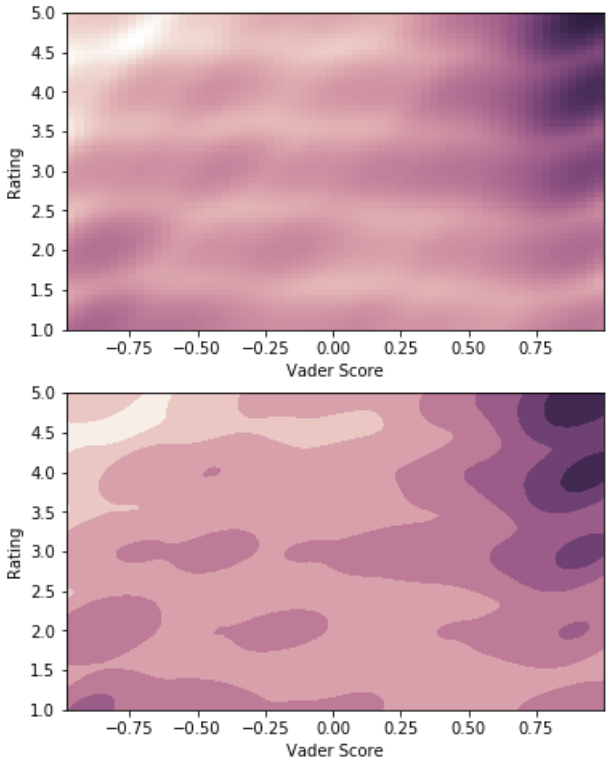
```
from scipy.stats.kde import gaussian_kde

k = gaussian_kde(np.vstack([V, R]))
xi, yi = np.mgrid[V.min():V.max():V.size**0.5*1j,R.min():R.max():R.size**0.5*1j]
zi = k(np.vstack([xi.flatten(), yi.flatten()]))
```

In [119]:

```
cmap = sns.cubehelix_palette(light=1, as_cmap=True)
fig = plt.figure(figsize=(6,8))
ax1 = fig.add_subplot(211)
ax2 = fig.add_subplot(212)

ax1.pcolormesh(xi, yi, np.log10(zi.reshape(xi.shape)), cmap=cmap)
ax2.contourf(xi, yi, np.log10(zi.reshape(xi.shape)), cmap=cmap)

ax1.set_xlim(V.min(), V.max())
ax1.set_ylim(R.min(), R.max())
ax2.set_xlim(V.min(), V.max())
ax2.set_ylim(R.min(), R.max())

ax1.set_xlabel('Vader Score')
ax1.set_ylabel('Rating')

ax2.set_xlabel('Vader Score')
ax2.set_ylabel('Rating')
```

Out[119]:

```
Text(0, 0.5, 'Rating')
```



The scatter plot of the ground truth versus vader score was not particularly informative on it own. The only interesting part about it was the for the higher ground through rating, the reviews (dots) were placed in the higher spectrum of the vader score. This idea in combination with the heatplot conveys the same idea, as the top right corner of the graph is a lot darker than the rest of the plot, which goes to show that those ratings had a higher vaderscore and a higher rating. And since we had established earlier, that most of the reviews had a higher vader score, the right half of the heat map is darker as that is the region with the most over lap between the high scores.

(b) Scatterplots and two heatmaps of the length of reviews versus each of ground truth score and Vader sentiment score. Each review is a point on the scatterplot.

In [120]:

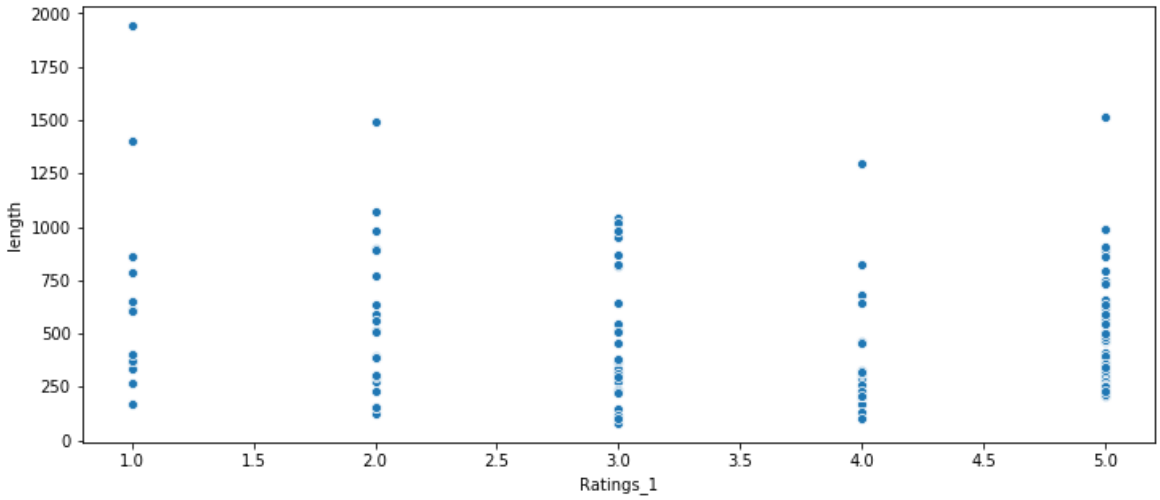
```
review = scatterDF['reviewCol']
length = []

for r in review:
    length.append(len(r))
length_df = pd.Series(length)
scatterDF1 = pd.concat([scatterDF,length_df.rename('length')],axis = 1)
scatterDF1.head()
```

Out[120]:

	reviewCol	vader	Experience	Hotel_Name	exp_index	Ratings_1	Ratings_2	length
0	"I was looking for a place to sit and chill fo...	0.8442	positive	The Riding Fool Hostel	1	NaN	5	439
1	"I stayed at the Riding Fool Hostel whilst I w...	0.9965	positive	The Riding Fool Hostel	1	NaN	5	2666
2	"My husband and I (both in our 50\\'s) stayed ...	0.9757	positive	The Riding Fool Hostel	1	NaN	5	509
3	"We were warmly welcomed by Caitlin, the new ...	0.9766	positive	The Riding Fool Hostel	1	NaN	5	673
4	"Comfortable, cheap, and cosy accommodation wi...	0.4939	positive	The Riding Fool Hostel	1	NaN	5	384

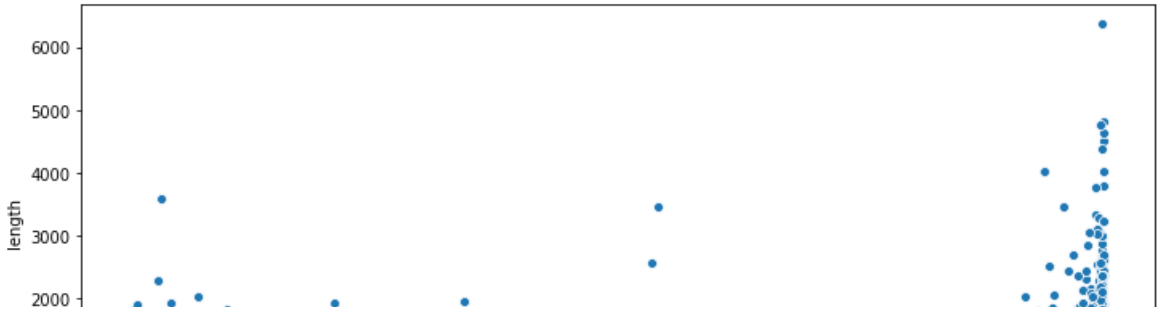
In [121]:

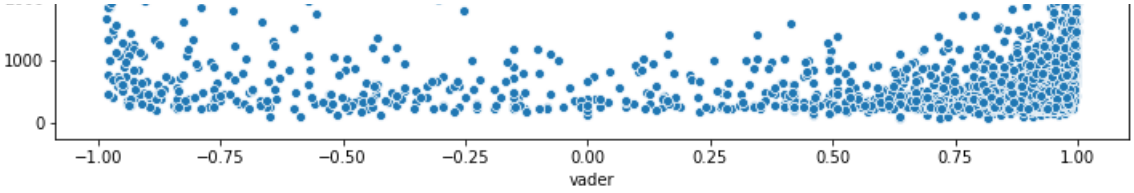
```
fig, ax = plt.subplots(figsize=(12,5))
ax = sns.scatterplot(x="Ratings_1", y="length", data=scatterDF1)
```



In [122]:

```
fig, ax = plt.subplots(figsize=(12,5))
ax = sns.scatterplot(x="vader", y="length", data=scatterDF1)
```





In [123]:

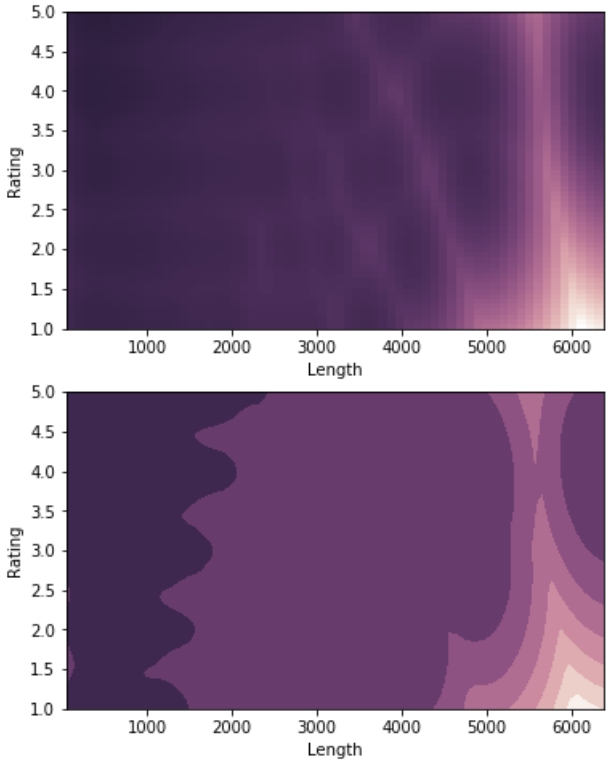
```
x = scatterDF1['length']
y = scatterDF1['Ratings_2']
k = gaussian_kde(np.vstack([x, y]))
xi, yi = np.mgrid[x.min():x.max():x.size**0.5*1j,y.min():y.max():y.size**0.5*1j]
zi = k(np.vstack([xi.flatten(), yi.flatten()]))
```

In [124]:

```
cmap = sns.cubehelix_palette(light=1, as_cmap=True)
fig = plt.figure(figsize=(6,8))
ax1 = fig.add_subplot(211)
ax2 = fig.add_subplot(212)

ax1.pcolormesh(xi, yi, np.log10(zi.reshape(xi.shape)), cmap=cmap)
ax2.contourf(xi, yi, np.log10(zi.reshape(xi.shape)), cmap=cmap)

ax1.set_xlim(x.min(), x.max())
ax1.set_ylim(y.min(), y.max())
ax2.set_xlim(x.min(), x.max())
ax2.set_ylim(y.min(), y.max())

ax1.set_xlabel('Length')
ax1.set_ylabel('Rating')

ax2.set_xlabel('Length')
ax2.set_ylabel('Rating')
```

Out[124]:

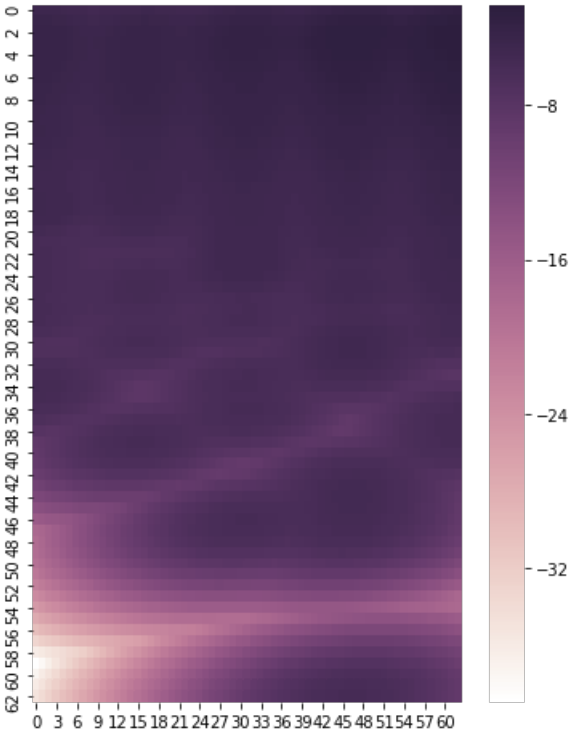
```
Text(0, 0.5, 'Rating')
```



In [125]:

```
fig = plt.figure(figsize=(6,8))
```

```
ax3 = sns.heatmap(np.log10(zi.reshape(xi.shape)),cmap=cmap)
```



The scatterplot tells us that most reviews are at a lower character length (about 2000 characters or lower) and this is reflected in the heat map, as most the color becomes lighter past review length of 1500 characters. Further more, there are a few tweets of character length 4000 or higher and this is reflected in the small light patch along that character length and review rating of 4.

In [126]:

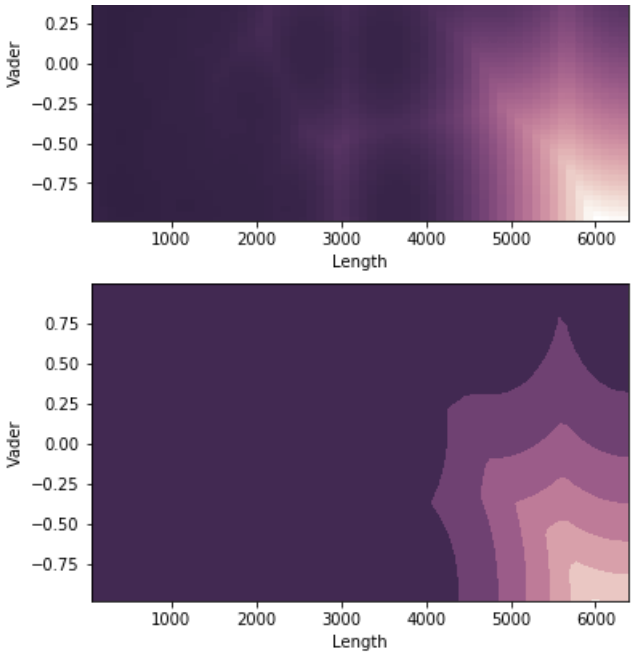
```
x = scatterDF1['length']
y = scatterDF1['vader']
k = gaussian_kde(np.vstack([x, y]))
xi, yi = np.mgrid[x.min():x.max():x.size**0.5*1j,y.min():y.max():y.size**0.5*1j]
zi = k(np.vstack([xi.flatten(), yi.flatten()]))
```

In [127]:

```
cmap = sns.cubehelix_palette(light=1, as_cmap=True)
fig = plt.figure(figsize=(6,8))
ax1 = fig.add_subplot(211)
ax2 = fig.add_subplot(212)

ax1.pcolormesh(xi, yi, np.log10(zi.reshape(xi.shape)), cmap=cmap)
ax2.contourf(xi, yi, np.log10(zi.reshape(xi.shape)), cmap=cmap)

ax1.set_xlim(x.min(), x.max())
ax1.set_ylim(y.min(), y.max())
ax2.set_xlim(x.min(), x.max())
ax2.set_ylim(y.min(), y.max())

ax1.set_xlabel('Length')
ax1.set_ylabel('Vader')

ax2.set_xlabel('Length')
ax2.set_ylabel('Vader')
```

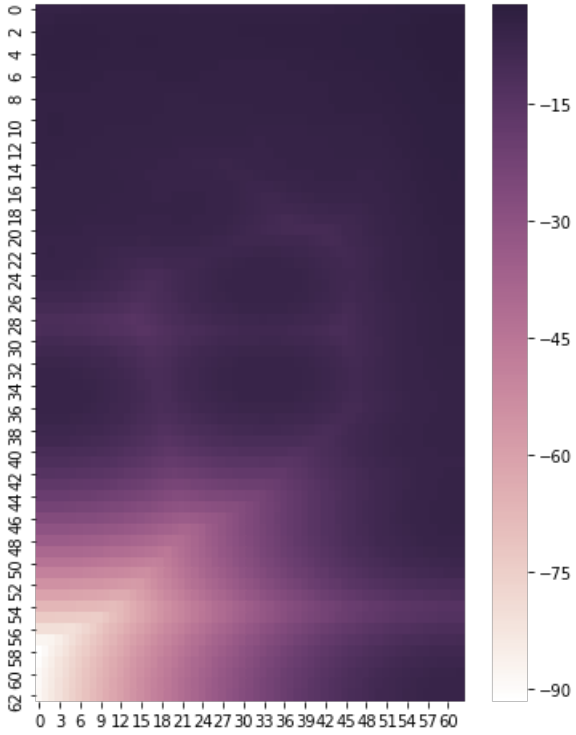
Out[127]:

```
Text(0, 0.5, 'Vader')
```



In [128]:

```
fig = plt.figure(figsize=(6,8))
ax3 = sns.heatmap(np.log10(zi.reshape(yi.shape)),cmap=cmap)
```



The scatter plot for the vader score was the opposite of the ratings where, the higher the score (more positive) a score, the more characters were used in the review. However the patterns of the vader score match that of the heat plot where the lower the score, the less number of characters were used in the review of the hotel.

(c) Scatterplots of the number of reviews per hotel versus each of average groundtruth score and average Vader sentiment score. In this case, each hotel is a single point on the scatterplot.

In [129]:

```
avg_sort =  avg_rating.sort_values(['vader'], ascending=[1])

x = list(avg_sort.index)
vad = list(avg_sort['vader'].values)
exp = list(avg_sort['exp_index'].values)
```

```
x_DF = pd.DataFrame(x)
df = pd.DataFrame({'average_vader':vad, 'average_exp':exp, 'HotelName': x})
```

In [130]:

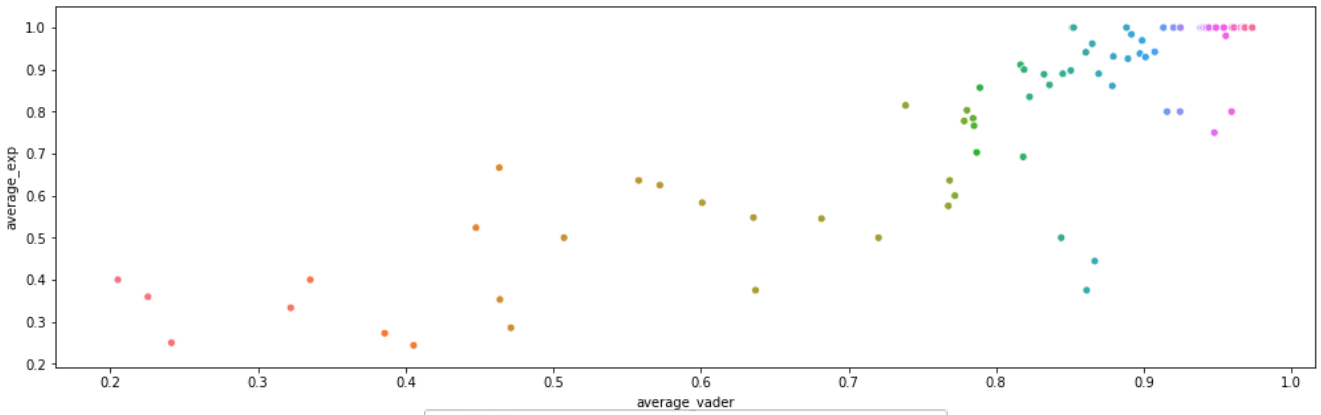
```
import pylab
```

In [131]:

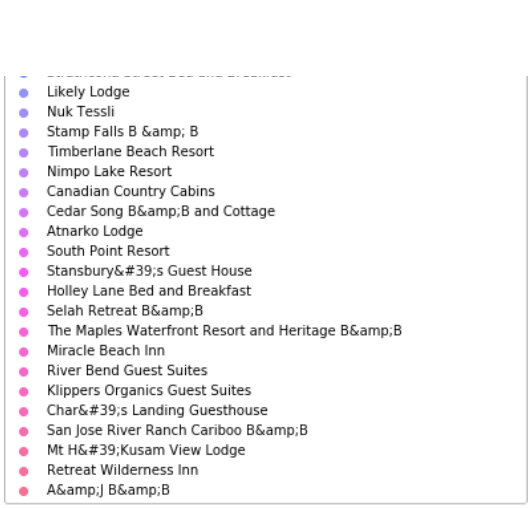
```
fig, ax = plt.subplots(figsize=(17,5))
ax = sns.scatterplot(x="average_vader", y="average_exp",hue = 'HotelName', data=df)
pylab.legend(loc=9, bbox_to_anchor=(0.5, -0.1))
```

Out[131]:

```
<matplotlib.legend.Legend at 0x1a32bbb710>
```



HotelName
Jewel Bay Resort
Abbotsford Hotel
Sorrento Inn
Tyee Village Motel
China Creek Campground
Bluebird Motel
Howard Johnson Hotel Port Alberni
A-1 Alberni Inn
Rangeland Motel
Twin Creeks Motel
Deerview Lodge & Cabins
Cumberland Lake Park Campground
Qualicum Bay Resort
New Caledonia Motel
The View Hotel
Sandcastle Inn
Sunset Motel and Trailer Park
Redford Motel & RV
Passages Resort & RV
Hummingbird Guesthouse
Causeway Bay Hotel Sparwood
Paradise Seashell Motel
Riverside Lodge B&B
Skylite Motel
Poco Inn & Suites Hotel
Sunrise Ridge Waterfront Resort
The Hospitality Inn
Quality Resort Bayside
The Riding Fool Hostel
Oceanside Village Resort
Tumbler Ridge Hotel and Suites
Holiday Inn Express & Suites Terrace
Best Western Plus Regency Inn & Conference Centre
Trend Mountain Hotel & Conference Centre
Sproat Lake Landing Resort
Timberlodge & RV Campground
Best Western Plus Barclay Hotel
Beach Club Resort - Bellstar Hotels & Resorts
Edelweiss Bed and Breakfast
Riverside Cabins
Stewart's Lodge & Camps
Valley Motel
Swept Away Inn
Tumbler Ridge Inn
Holiday Inn Express & Suites Victoria - Colwood
Somass Motel & RV
Beach Acres Resort
Artful Retreat B&B
Shuswap Lake Motel and Resort
Eagle Nook Resort
Arbutus Grove Motel
Riverside Motel
Sunwolf
Cedar Wood Lodge Bed & Breakfast Inn & Conference Center
Mozey-On-Inn
Ten-ee-ah Lodge
Strathcona Street Bed and Breakfast



There is a positive linear pattern between the average vader scores and the average ratings per hotel, which suggest that even if the two scores are not linearly connected,they are still however dependent on each other.

In [ ]: