hotel-reviews-analysis

May 7, 2023

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('ggplot')
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A
NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy
(detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

[2]:
```
data = pd.read_csv('../input/hotel-reviews/Datafiniti_Hotel_Reviews.csv')

# viewing the dataset
print(f'Rows: {data.shape[0]}, Columns: {data.shape[1]}')
data.columns
```

```
Rows: 10000, Columns: 25
```

[2]:
```
Index(['id', 'dateAdded', 'dateUpdated', 'address', 'categories',
       'primaryCategories', 'city', 'country', 'keys', 'latitude', 'longitude',
       'name', 'postalCode', 'province', 'reviews.date', 'reviews.dateSeen',
       'reviews.rating', 'reviews.sourceURLs', 'reviews.text', 'reviews.title',
       'reviews.userCity', 'reviews.userProvince', 'reviews.username',
       'sourceURLs', 'websites'],
      dtype='object')
```

0.1 Preparing the data to analzye

[3]:
```
data = data[['id', 'categories', 'city', 'reviews.date', 'reviews.rating',␣
 ↪'reviews.text', 'websites']]
data.head(3)
```

[3]:

	id	categories	\
0	AVwc252WIN2L1WUfpqLP	Hotels,Hotels and motels,Hotel and motel reser…	
1	AVwc252WIN2L1WUfpqLP	Hotels,Hotels and motels,Hotel and motel reser…	
2	AVwc252WIN2L1WUfpqLP	Hotels,Hotels and motels,Hotel and motel reser…	

```
              city          reviews.date  reviews.rating  \
0  Rancho Santa Fe  2013-11-14T00:00:00Z             5.0
1  Rancho Santa Fe  2014-07-06T00:00:00Z             5.0
2  Rancho Santa Fe  2015-01-02T00:00:00Z             5.0

                                        reviews.text  \
0  Our experience at Rancho Valencia was absolute…
1  Amazing place. Everyone was extremely warm and…
2  We booked a 3 night stay at Rancho Valencia to…

                       websites
0  http://www.ranchovalencia.com
1  http://www.ranchovalencia.com
2  http://www.ranchovalencia.com
```

0.1.1 Extracting *Hotel name* from website link

[4]:
```
data['websites'] = data['websites'].str.split('.').str[1]
data.rename(columns={'websites': 'Hotel'}, inplace=True)
```

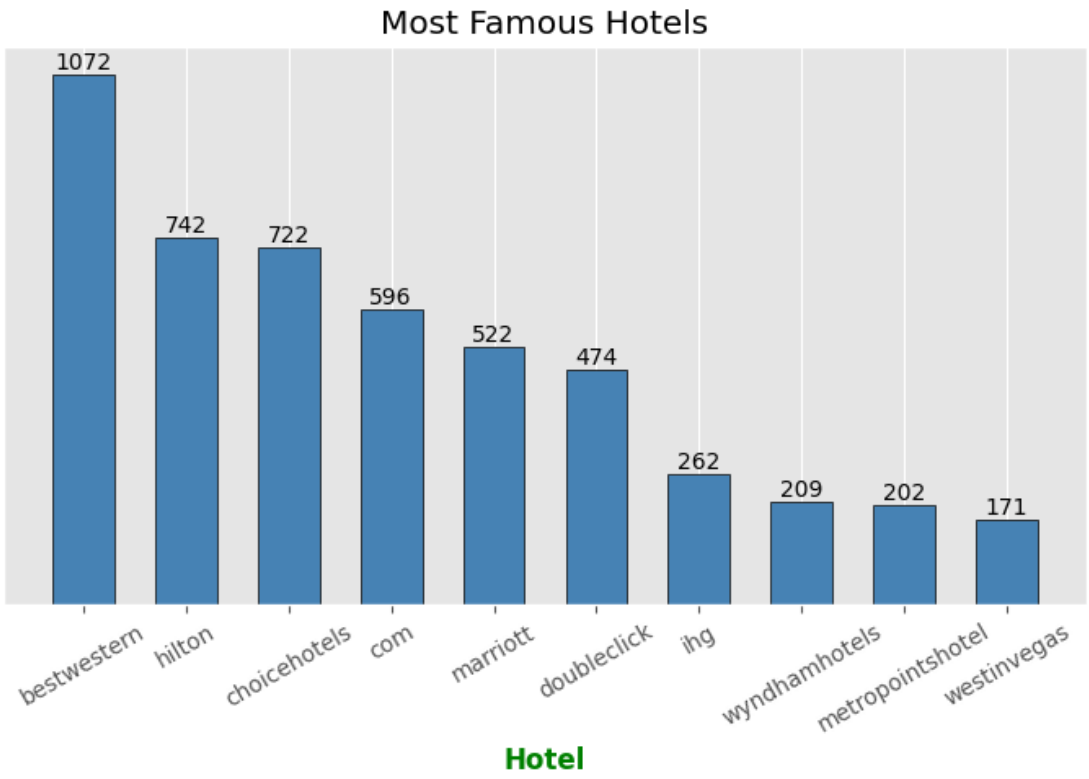
0.2 Tracking Most famous hotels

[5]:
```
hotels= data['Hotel'].value_counts().head(10)
hotels
```

[5]:
```
bestwestern         1072
hilton               742
choicehotels         722
com                  596
marriott             522
doubleclick          474
ihg                  262
wyndhamhotels        209
metropointshotel     202
westinvegas          171
Name: Hotel, dtype: int64
```

[6]:
```
# Create a figure with custom size
fig = plt.figure(figsize=(7, 5))

# Create a bar plot
plt.bar(hotels.index, hotels.values, color='steelblue', width=0.6,␣
 ↪edgecolor='black')

# Add value labels to the bars
for i, v in enumerate(hotels.values):
```

```
    plt.text(i, v, str(v), ha='center', va='bottom')

# Set the x-label with customized font properties
plt.xlabel('Hotel', fontdict={'fontsize': 12, 'fontweight': 'bold', 'color':␣
 ↪'green'})
plt.xticks(rotation=30, fontsize=10)
plt.yticks([])


plt.title('Most Famous Hotels')

# Add grid lines
plt.grid(True)

# Adjust plot spacing
plt.tight_layout()

# Display the plot
plt.show()
```



0.3 Doing sentimenal analysis on our data

-

0.3.1 Using Roberta pretrained model to do this

[7]:
```
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
from tqdm.notebook import tqdm
```

[8]:
```
MODEL = f'cardiffnlp/twitter-roberta-base-sentiment'
tokenizer = AutoTokenizer.from_pretrained(MODEL)
model = AutoModelForSequenceClassification.from_pretrained(MODEL)
```

```
Downloading (…)lve/main/config.json:   0%|          | 0.00/747 [00:00<?, ?B/s]
Downloading (…)olve/main/vocab.json:   0%|          | 0.00/899k [00:00<?, ?B/s]
Downloading (…)olve/main/merges.txt:   0%|          | 0.00/456k [00:00<?, ?B/s]
Downloading (…)cial_tokens_map.json:   0%|          | 0.00/150 [00:00<?, ?B/s]
Downloading pytorch_model.bin:   0%|          | 0.00/499M [00:00<?, ?B/s]
```

0.3.2 Assigning sentimental score to reviews

[9]:
```
# function to assign sentimental scores to text
def polarity_scores_roberta(example):
    encoded_text = tokenizer(example, return_tensors='pt')
    output = model(**encoded_text)
    scores = output[0][0].detach().numpy()
    scores = softmax(scores)

    scores_dict = {
        'roberta_neg': scores[0],
        'roberta_neu': scores[1],
        'roberta_pos': scores[2]
    }

    return scores_dict
```

[10]:
```
# will run test only on top 500 "Most Famous Hotels"
filtered_hotels = data['Hotel'].isin(hotels.index)
famous_hotels = data[filtered_hotels].head(500)

result = dict()
for i, row in tqdm(famous_hotels.iterrows(), total=len(famous_hotels)):
    try:
        text = row['reviews.text']
```

```
        myid = row['id']

        roberta_result = polarity_scores_roberta(text)
        result[myid] = roberta_result
    except Exception as error:
        print(f'{error} occured at {myid} id')
```

```
  0%|          | 0/500 [00:00<?, ?it/s]
```

```
The expanded size of the tensor (670) must match the existing size (514) at non-
singleton dimension 1.  Target sizes: [1, 670].  Tensor sizes: [1, 514] occured
at AV6ZfnuiRxPSIh2RlHtM id
The expanded size of the tensor (650) must match the existing size (514) at non-
singleton dimension 1.  Target sizes: [1, 650].  Tensor sizes: [1, 514] occured
at AVwdIPxpkufWRAb52lqx id
```

[11]:
```
sentimented_df = pd.DataFrame(result).T
sentimented_df = sentimented_df.reset_index().rename(columns={'index': 'id'})
sentimented_df = sentimented_df.merge(data, how='left')

sentimented_df.head(2)
```

[11]:

	id	roberta_neg	roberta_neu	roberta_pos	\
0	AVwePiAX_7pvs4fzBSAl	0.001499	0.008289	0.990213	
1	AVwePiAX_7pvs4fzBSAl	0.001499	0.008289	0.990213	

	categories	city	\
0	Hotels,Hotels and motels,Hotel and motel reser…	Vancouver	
1	Hotels,Hotels and motels,Hotel and motel reser…	Vancouver	

	reviews.date	reviews.rating	\
0	2016-01-26T00:00:00Z	5.0	
1	2016-05-03T00:00:00Z	5.0	

	reviews.text	Hotel
0	In my line of work, I use meeting space in hot…	hilton
1	The staff is very friendly and helpful. The ro…	hilton

0.3.3 Doing sentimental analysis over time

[12]:
```
# first need to extract year from "reviews.date"
sentimented_df['reviews.date'] = sentimented_df['reviews.date'].str[:4]
sentimented_df.head(2)
```

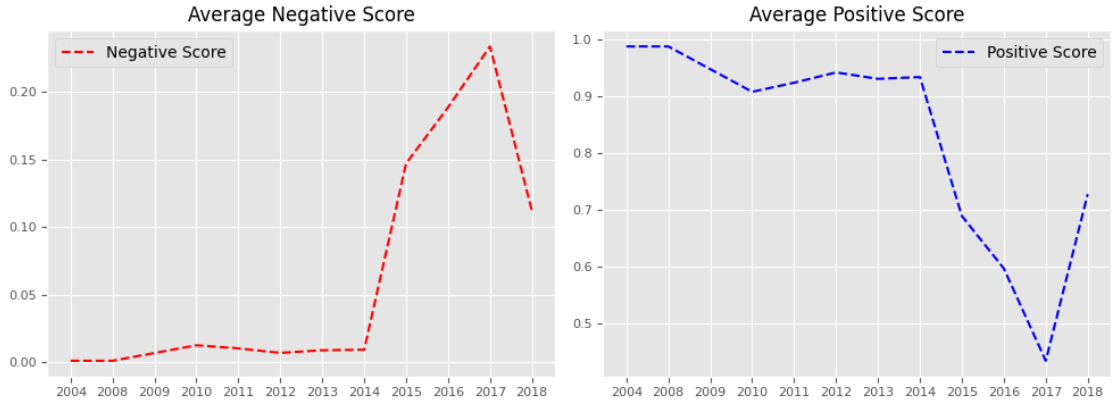
[12]:

	id	roberta_neg	roberta_neu	roberta_pos	\
0	AVwePiAX_7pvs4fzBSAl	0.001499	0.008289	0.990213	
1	AVwePiAX_7pvs4fzBSAl	0.001499	0.008289	0.990213	

```
                                          categories       city reviews.date  \
0  Hotels,Hotels and motels,Hotel and motel reser…  Vancouver         2016
1  Hotels,Hotels and motels,Hotel and motel reser…  Vancouver         2016

   reviews.rating                                       reviews.text   Hotel
0             5.0  In my line of work, I use meeting space in hot…  hilton
1             5.0  The staff is very friendly and helpful. The ro…  hilton
```

[13]:
```
# we would be grouping the data
grp_by_year = sentimented_df.groupby('reviews.date')

# extract the average positive, negative 'roberta_sentimented score'
avg_negative_score = grp_by_year['roberta_neg'].mean()
avg_positive_score = grp_by_year['roberta_pos'].mean()
```

[14]:
```
# showing the trend through visuals
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

sns.lineplot(data=avg_negative_score, x=avg_negative_score.index,␣
 ↪y=avg_negative_score.values, ax=axes[0], color='red', linestyle='dashed')
sns.lineplot(data=avg_positive_score, x=avg_positive_score.index,␣
 ↪y=avg_positive_score.values, ax=axes[1], color='blue', linestyle='dashed')

axes[0].grid(True)  # Add grid lines to the first subplot
axes[1].grid(True)  # Add grid lines to the second subplot

axes[0].set_xlabel(' '); axes[1].set_xlabel(' ') # hide the x-axis labels for␣
 ↪both visuals

axes[0].legend(['Negative Score'], loc='upper left')  # Add a legend for the␣
 ↪first subplot
axes[1].legend(['Positive Score'], loc='upper right')  # Add a legend for the␣
 ↪second subplot

axes[0].tick_params(labelsize=8)  # Increase tick label font size in the first␣
 ↪subplot
axes[1].tick_params(labelsize=8)  # Increase tick label font size in the second␣
 ↪subplot

axes[0].set_title('Average Negative Score', font='Seoge UI Bold', fontsize=12)
axes[1].set_title('Average Positive Score', font='Seoge UI Bold', fontsize=12)

plt.style.use('ggplot')
plt.tight_layout()
plt.show()
```



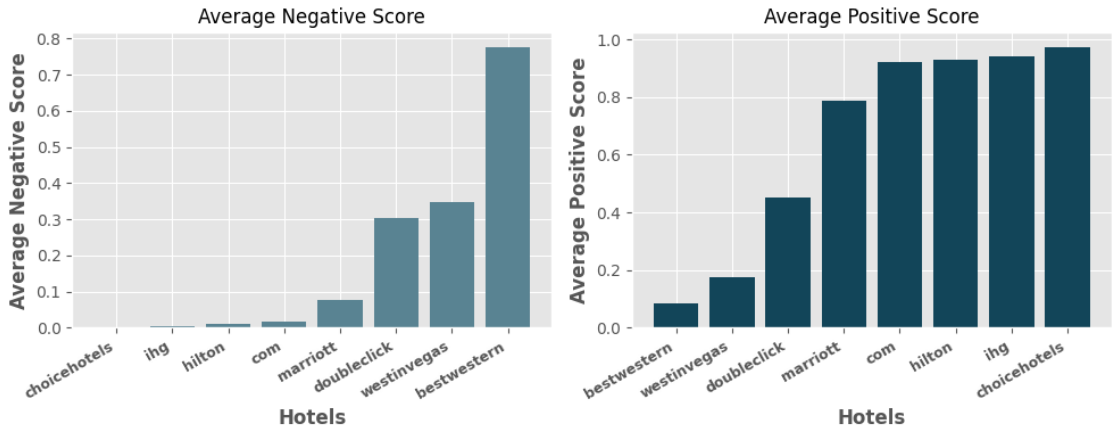
0.3.4 Let's check for which hotels have gotten most good and bad reviews

[15]:
```
grp_by_hotels = sentimented_df.groupby('Hotel')

# extracting the average negative & positive score
avg_neg_score = grp_by_hotels['roberta_neg'].mean()
avg_pos_score = grp_by_hotels['roberta_pos'].mean()

avg_neg_score = avg_neg_score.reset_index().sort_values(by='roberta_neg')
avg_pos_score = avg_pos_score.reset_index().sort_values(by='roberta_pos')
```

[16]:
```
import matplotlib.pyplot as plt

fig, axes = plt.subplots(1, 2, figsize=(10, 4))

# Bar plot for Average Negative Score
axes[0].bar(avg_neg_score['Hotel'], avg_neg_score['roberta_neg'],␣
 ↪color='#598392')
axes[0].grid(True)  # Add grid lines to the first subplot
axes[0].set_xlabel('Hotels', fontsize=12, fontweight='bold')
axes[0].set_ylabel('Average Negative Score', fontsize=12, fontweight='bold')
axes[0].set_xticklabels(avg_neg_score['Hotel'], rotation=30, ha='right',␣
 ↪fontsize=10, fontweight='bold')
axes[0].tick_params(axis='x', labelsize=9, rotation=30)  # Increase tick label␣
 ↪font size in the first subplot
axes[0].set_title('Average Negative Score', font='Seoge UI Bold', fontsize=12)

# Bar plot for Average Positive Score
axes[1].bar(avg_pos_score['Hotel'], avg_pos_score['roberta_pos'],␣
 ↪color='#124559')
axes[1].grid(True)  # Add grid lines to the second subplot
```

```
axes[1].set_xlabel('Hotels', fontsize=12, fontweight='bold')
axes[1].set_ylabel('Average Positive Score', fontsize=12, fontweight='bold')
axes[1].set_xticklabels(avg_pos_score['Hotel'], rotation=30, ha='right',␣
 ↪fontsize=10, fontweight='bold')
axes[1].tick_params(axis='x', rotation=30, labelsize=9)
axes[1].set_title('Average Positive Score', fontsize=12)

plt.style.use('ggplot')
plt.tight_layout()
plt.show()
```

```
/tmp/ipykernel_20/4135284405.py:10: UserWarning: FixedFormatter should only be
used together with FixedLocator
  axes[0].set_xticklabels(avg_neg_score['Hotel'], rotation=30, ha='right',
fontsize=10, fontweight='bold')
/tmp/ipykernel_20/4135284405.py:19: UserWarning: FixedFormatter should only be
used together with FixedLocator
  axes[1].set_xticklabels(avg_pos_score['Hotel'], rotation=30, ha='right',
fontsize=10, fontweight='bold')
```



0.3.5 Check out for how many reviews are *negative* or *positive*

[17]:
```
import numpy as np

sentimented_df['review.type'] = np.where(sentimented_df['roberta_pos'] >␣
 ↪sentimented_df['roberta_neg'],
                                         'Positive', 'Negative')
sentimented_df.head(2)
```

[17]:

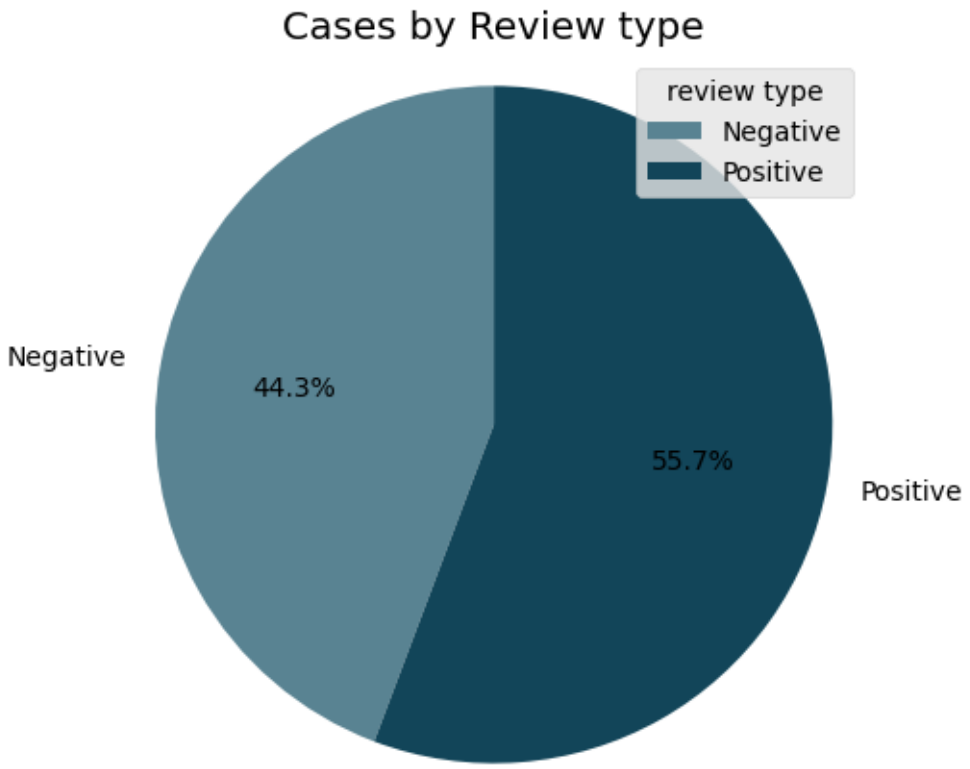
	id	roberta_neg	roberta_neu	roberta_pos	\
0	AVwePiAX_7pvs4fzBSAl	0.001499	0.008289	0.990213	

```
1  AVwePiAX_7pvs4fzBSAl     0.001499     0.008289     0.990213

                                          categories       city reviews.date  \
0  Hotels,Hotels and motels,Hotel and motel reser…  Vancouver         2016
1  Hotels,Hotels and motels,Hotel and motel reser…  Vancouver         2016

   reviews.rating                                    reviews.text   Hotel  \
0             5.0  In my line of work, I use meeting space in hot…  hilton
1             5.0  The staff is very friendly and helpful. The ro…  hilton

  review.type
0    Positive
1    Positive
```

[18]:
```
grp_by_reviewtype = sentimented_df.groupby('review.type')
total_reviews = grp_by_reviewtype.size()

total_reviews
```

[18]:
```
review.type
Negative    231
Positive    291
dtype: int64
```

[19]:
```
# set a custom color palette
colors = ['#598392', '#124559']

# create a figure and axis object
fig, ax = plt.subplots(figsize=(5, 5))

# create a pie chart
ax.pie(total_reviews, labels=total_reviews.index, colors=colors, autopct='%1.
 ↪1f%%', startangle=90)

# add a title and legend
ax.set_title('Cases by Review type')
ax.legend(title='review type', loc='best')

# make the pie chart circular
ax.axis('equal')

# display the plot
plt.show()
```



0.3.6 Check the average negative & positive score by cities

[20]:
```
# check for what are the most famous cities for people to stay in hotels
famous_cities = sentimented_df['city'].value_counts().head(10)
famous_cities
```

[20]:
```
Las Vegas       173
Boston          100
Phoenix          40
Anaheim          38
Atlanta          38
Oxon Hill        31
Napa             25
Key West         20
Philadelphia     18
Chicago          14
Name: city, dtype: int64
```

[21]:
```
# filter the data for most famous cities and then group it
filtered_data = sentimented_df['city'].isin(famous_cities.index)
famous_cities = sentimented_df[filtered_data]
```

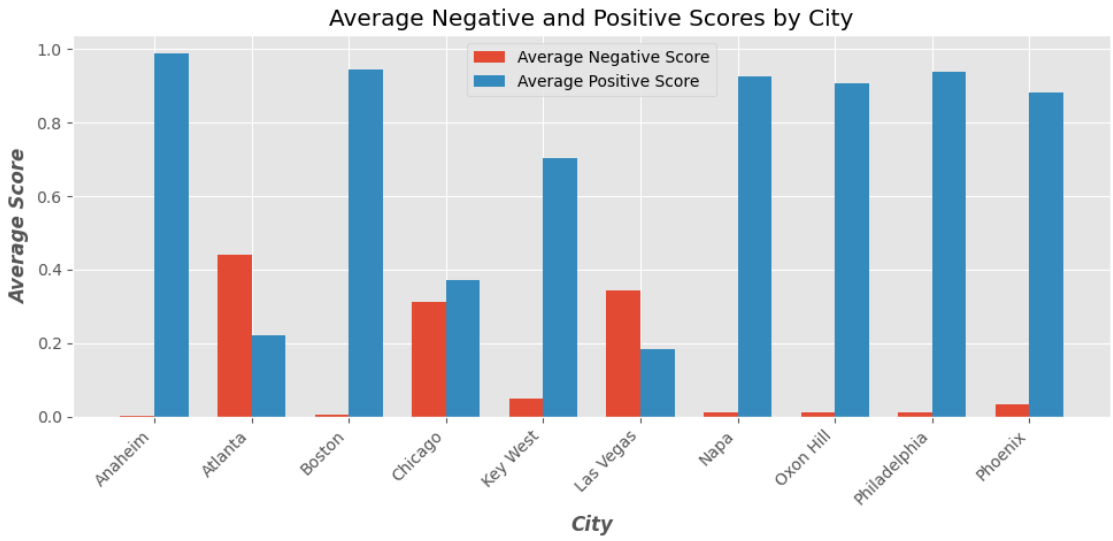
```
grp_by_city = famous_cities.groupby('city')
grp_by_city = grp_by_city[['roberta_neg', 'roberta_pos']].mean()
grp_by_city.reset_index(inplace=True)

grp_by_city.head(5)
```

[21]:

	city	roberta_neg	roberta_pos
0	Anaheim	0.001251	0.987836
1	Atlanta	0.441629	0.221839
2	Boston	0.005276	0.945432
3	Chicago	0.311736	0.371368
4	Key West	0.049348	0.703882

[22]:
```
# Set the figure size
plt.figure(figsize=(10, 5))

# Get the number of cities
num_cities = len(grp_by_city)

# Set the width for each bar
bar_width = 0.35

# Set the index for the x-axis ticks
index = np.arange(num_cities)

# Plot the average negative scores
plt.bar(index, grp_by_city['roberta_neg'], width=bar_width, label='Average␣
 ↪Negative Score')

# Plot the average positive scores
plt.bar(index + bar_width, grp_by_city['roberta_pos'], width=bar_width,␣
 ↪label='Average Positive Score')

# Set the x-axis tick labels
plt.xticks(index + bar_width/2, grp_by_city['city'], rotation=45, ha='right')

# Set the labels and title
plt.xlabel('City', fontdict={'family': 'sans-serif', 'size': 12, 'weight':␣
 ↪'bold', 'style': 'italic'})
plt.ylabel('Average Score', fontdict={'family': 'sans-serif', 'size': 12,␣
 ↪'weight': 'bold', 'style': 'italic'})
plt.title('Average Negative and Positive Scores by City')

# Add a legend
legend_font = {'size': 10}
plt.legend(loc='best', prop=legend_font)
# Show the plot
```

```
plt.style.use('fivethirtyeight')
# plt.style.use('ggplot')
plt.tight_layout()
plt.show()
```



1 Insights from the analysis

- People likeliness about hotels has been decreasing over time but lately it seems like it has been coming back on track
- People tend to go to places what has left good impression on others. Well, it is right for most of the hotels but bestwestern is showing a complete different story even after getting the most customers, it has left the worst impression
- Positive reviews about their hotel experience by people are slightly higher than negative ones but it can catch on if hotel management doesn't do anything about it in next coming years
- Las Vegas is showing the same story as bestwestern, even after being the most popular place to stay in hotels, it hasn't been able to left good impression on those people.

1.0.1 Well this is all I have found from this dataset, there is much more you could do using this dataset. So, try it as you like