**Yelp Reviews: Sentiment Analysis**

In [1]:

**import** **pandas** **as** **pd**  
**import** **numpy** **as** **np**  
**import** **matplotlib.pyplot** **as** **plt**  
%**matplotlib** inline  
**import** **seaborn** **as** **sns**  
**import** **nltk**  
**from** **nltk.corpus** **import** stopwords

**Loading Dataset into Pandas DataFrame**[**¶**](#gjdgxs)

In [2]:

dataset=pd.read\_csv("yelp\_reviews.csv")

In [3]:

dataset.head(2)

Out[3]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **address** | **alias** | **avg\_rating** | **city** | **country** | **physician** | **rating** | **state** | **text** | **time\_created** | **title** | **url** | **user** | **zip\_code** |
| **0** | 35 E 21st St, Fl 7, New York, NY 10010 | internalmed | 4.5 | New York | US | Laura Guderian, MD | 5 | NY | I've been delighted with Dr. Guderian as my pr... | 2017-09-06 13:21:59 | Internal Medicine | https://www.yelp.com/biz/laura-guderian-md-new... | Bonnie E. | 10010 |
| **1** | 35 E 21st St, Fl 7, New York, NY 10010 | internalmed | 4.5 | New York | US | Laura Guderian, MD | 1 | NY | I really wanted to like One Medical Group. I a... | 2016-09-25 16:00:50 | Internal Medicine | https://www.yelp.com/biz/laura-guderian-md-new... | Yelpish N. | 10010 |

In [6]:

dataset.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 145 entries, 0 to 144  
Data columns (total 14 columns):  
address 145 non-null object  
alias 145 non-null object  
avg\_rating 145 non-null float64  
city 145 non-null object  
country 145 non-null object  
physician 145 non-null object  
rating 145 non-null int64  
state 145 non-null object  
text 145 non-null object  
time\_created 145 non-null object  
title 145 non-null object  
url 145 non-null object  
user 145 non-null object  
zip\_code 145 non-null int64  
dtypes: float64(1), int64(2), object(11)  
memory usage: 15.9+ KB

In [7]:

dataset.describe()

Out[7]:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **avg\_rating** | **rating** | **zip\_code** |
| **count** | 145.000000 | 145.000000 | 145.000000 |
| **mean** | 4.375862 | 4.062069 | 10393.296552 |
| **std** | 0.708409 | 1.560007 | 757.288558 |
| **min** | 2.000000 | 1.000000 | 7030.000000 |
| **25%** | 4.000000 | 3.000000 | 10016.000000 |
| **50%** | 4.500000 | 5.000000 | 10028.000000 |
| **75%** | 5.000000 | 5.000000 | 11201.000000 |
| **max** | 5.000000 | 5.000000 | 11416.000000 |

**Text Length to represent How skewed the data is!**[**¶**](#30j0zll)

Reviews with maximum rating have longer length in comparison to reviews with less ratings

In [8]:

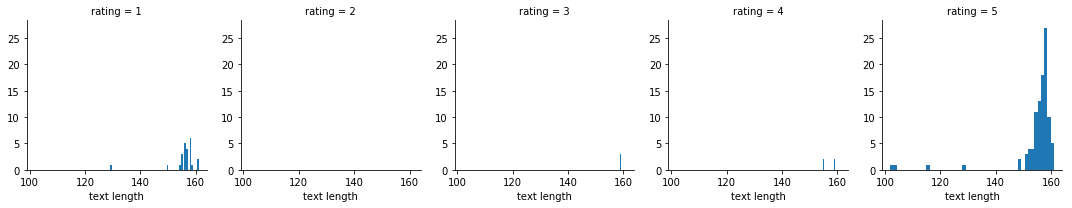
*#To get an insight on the length of each review, we can create a new column in yelp called text length*  
dataset['text length'] = dataset['text'].apply(len)

In [10]:

*#Exploring the dataset*  
g = sns.FacetGrid(data=dataset, col='rating')  
g.map(plt.hist, 'text length', bins=50)  
*#We can see thet there are high number of 5-star reviews*

Out[10]:

<seaborn.axisgrid.FacetGrid at 0x1ef21ab79b0>



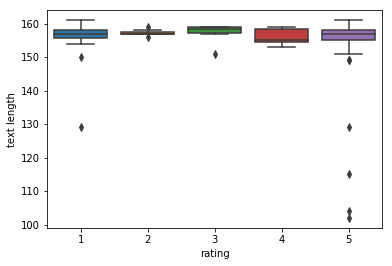
**Outlier Detection**[**¶**](#1fob9te)

In [11]:

*#As we can say that rating 1 ,2,3,4,5 are skewed towards right now we will se about outliers using box plots*  
sns.boxplot(x='rating', y='text length', data=dataset)  
*#as we can say there are outliers so text length as may not be as a best feature for predicting*

Out[11]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x1ef21ed6710>



In [12]:

*#Finding the correlation*  
stars = dataset.groupby('rating').mean()  
stars.corr()

Out[12]:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **avg\_rating** | **zip\_code** | **text length** |
| **avg\_rating** | 1.000000 | 0.467536 | -0.459738 |
| **zip\_code** | 0.467536 | 1.000000 | -0.356836 |
| **text length** | -0.459738 | -0.356836 | 1.000000 |

**Correlation between Average Rating, ZIP Code and Length of Reviews**[**¶**](#3znysh7)

In [13]:

sns.heatmap(data=stars.corr(), annot=**True**)

Out[13]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x1ef225fe198>



**Creating Classes as Positive for 5 rating and Negative for 1 rating**[**¶**](#2et92p0)

In [14]:

*#Our task is to predict either the review is good or bad so just grab reviews that are either 1 or 5*   
yelp\_class = dataset[(dataset['rating'] == 1) | (dataset['rating'] == 5)]

In [15]:

yelp\_class.shape

Out[15]:

(125, 15)

In [16]:

*# as we can see there are 125 rows, this is because*   
*# we are not taking 2,3,4 rating into account becuase much data for these ratings are not available*  
*#Let create x and y column for classification on the basis of text*  
X = yelp\_class['text']  
y = yelp\_class['rating']

In [17]:

*#Text Preprocessing*  
**import** **string**  
  
**def** text\_process(text):  
  
 *'''*  
 *Takes in a string of text, then performs the following:*  
 *1. Remove all punctuation*  
 *2. Remove all stopwords*  
 *3. Return the cleaned text as a list of words*  
 *'''*  
  
 nopunc = [char **for** char **in** text **if** char **not** **in** string.punctuation]  
  
 nopunc = ''.join(nopunc)  
   
 **return** [word **for** word **in** nopunc.split() **if** word.lower() **not** **in** stopwords.words('english')]

In [18]:

*#Testing on a sample text*  
sample\_text = "Hey there! This is a sample review, which happens to contain punctuations."  
  
print(text\_process(sample\_text))

['Hey', 'sample', 'review', 'happens', 'contain', 'punctuations']

**Creating Bag of Words Transformer using Count Vectorizer**[**¶**](#tyjcwt)

In [19]:

**from** **sklearn.feature\_extraction.text** **import** CountVectorizer  
count\_vect = CountVectorizer(analyzer=text\_process)  
bow\_transformer = count\_vect.fit(X) *# Bag of Words Transformer*

In [20]:

len(bow\_transformer.vocabulary\_)

Out[20]:

887

**A sample Review**[**¶**](#3dy6vkm)

In [21]:

review\_2 = X[1]

In [22]:

review\_2

Out[22]:

"I really wanted to like One Medical Group. I am a terrible patient (I don't like being poisoned or abused.) and have had major problems in the past with..."

In [23]:

bow\_2 = bow\_transformer.transform([review\_2])  
bow\_2

Out[23]:

<1x887 sparse matrix of type '<class 'numpy.int64'>'  
 with 14 stored elements in Compressed Sparse Row format>

In [24]:

X = bow\_transformer.transform(X)

**Splitting Training and Testing Dataset**[**¶**](#1t3h5sf)

In [25]:

*#Training and testing dataset*  
**from** **sklearn.model\_selection** **import** train\_test\_split  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=101)

**Fitting Data into Multinomila Naive Bayes Model for Sentiment Analysis**[**¶**](#4d34og8)

In [26]:

**from** **sklearn.naive\_bayes** **import** MultinomialNB  
  
nb = MultinomialNB()  
nb.fit(X\_train, y\_train)

Out[26]:

MultinomialNB(alpha=1.0, class\_prior=None, fit\_prior=True)

In [27]:

preds = nb.predict(X\_test)

**Classification Report of Sentiment Analysis**[**¶**](#2s8eyo1)

In [28]:

**from** **sklearn.metrics** **import** confusion\_matrix, classification\_report  
  
print(confusion\_matrix(y\_test, preds))  
print('**\n**')  
print(classification\_report(y\_test, preds))

[[ 2 7]  
 [ 6 23]]  
  
  
 precision recall f1-score support  
  
 1 0.25 0.22 0.24 9  
 5 0.77 0.79 0.78 29  
  
avg / total 0.64 0.66 0.65 38

**A positive Review Sample**[**¶**](#17dp8vu)

In [29]:

positive\_review = yelp\_class['text'][0]

In [30]:

positive\_review

Out[30]:

"I've been delighted with Dr. Guderian as my primary care doctor. She is thoughtful, attentive, and kind. She responds to my questions and concerns promptly,..."

**Predicting Rating for the Positive Sample Review**[**¶**](#3rdcrjn)

In [31]:

positive\_review\_transformed = bow\_transformer.transform([positive\_review])  
  
nb.predict(positive\_review\_transformed)[0]

Out[31]:

5