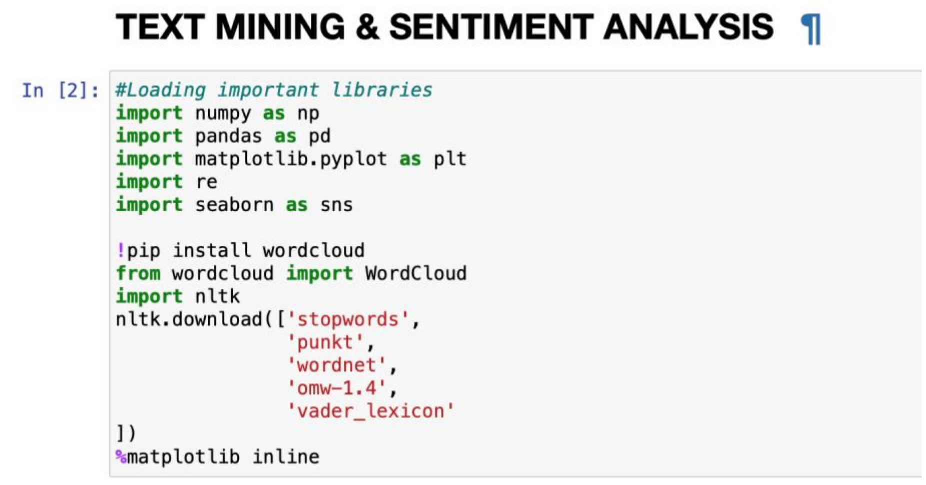
## Dataset

The dataset used for this study is the tourist accommodation reviews dataset. The source for this dataset is unknown.

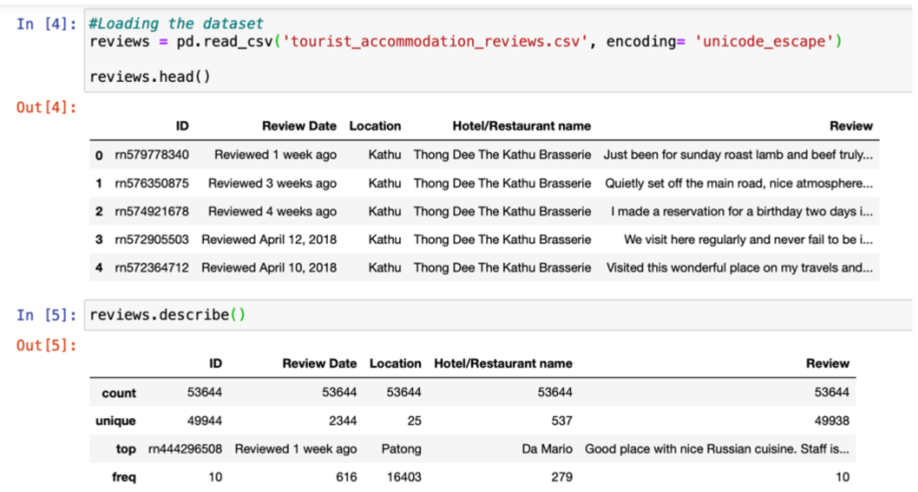
## Explanation and preparation of datasets

To see the dimensions and attributes of the dataset, we would be using python and its packages to understand the dataset.



### TEXT MINING & SENTIMENT ANALYSIS ¶

```
In [2]: #Loading important libraries
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import re
        import seaborn as sns

        !pip install wordcloud
        from wordcloud import WordCloud
        import nltk
        nltk.download(['stopwords',
                       'punkt',
                       'wordnet',
                       'omw-1.4',
                       'vader_lexicon'
        ])
        %matplotlib inline
```



```
In [4]: #Loading the dataset
        reviews = pd.read_csv('tourist_accommodation_reviews.csv', encoding= 'unicode_escape')

        reviews.head()
Out[4]:
             ID          Review Date  Location  Hotel/Restaurant name                                          Review
   0  rn579778340    Reviewed 1 week ago     Kathu  Thong Dee The Kathu Brasserie  Just been for sunday roast lamb and beef truly...
   1  rn576350875   Reviewed 3 weeks ago     Kathu  Thong Dee The Kathu Brasserie  Quietly set off the main road, nice atmosphere...
   2  rn574921678   Reviewed 4 weeks ago     Kathu  Thong Dee The Kathu Brasserie     I made a reservation for a birthday two days i...
   3  rn572905503  Reviewed April 12, 2018   Kathu  Thong Dee The Kathu Brasserie        We visit here regularly and never fail to be i...
   4  rn572364712  Reviewed April 10, 2018   Kathu  Thong Dee The Kathu Brasserie  Visited this wonderful place on my travels and...

In [5]: reviews.describe()
Out[5]:
                  ID          Review Date  Location  Hotel/Restaurant name                                         Review
   count       53644                53644     53644                  53644                                          53644
   unique      49944                 2344        25                    537                                          49938
      top rn444296508  Reviewed 1 week ago    Patong               Da Mario  Good place with nice Russian cuisine. Staff is...
     freq         10                  616     16403                    279                                             10
```

We can see that there are 53644 reviews in this dataset with 5 variables namely ID, Review date, Location, Hotel/Restaurant name and Review. For this study, we would be picking 30 Hotels/Restaurants. We will be making this selection based on Location. For this reason, we would be analysing the different locations. The table below summarises the location and their states and countries.

| LOCATION | STATE |
|---|---|
| Kathu | Yunnan, China |

| | |
|---|---|
| Kata Beach | Karon, Thailand |
| Rawai | Coast of Phuket Island, Thailand |
| Choeng Thale | Phuket province, Thailand |
| Karon Beach | Karon, Thailand |
| Phuket Town | Phuket Island, Thailand |
| Patong | Phuket Island, Thailand |
| Mai Khao | Phuket Island, Thailand |
| Karon | Karon, Thailand |
| Chalong | Phuket Island, Thailand |
| Nai Harn | Phuket, Thailand |
| Cape Panwa | Mueang Phuket District, Thailand |
| Sakhu | Phuket Island, Thailand |
| Pa Khlok | Pa Klok, Thailand |
| Kamala | Kamala, Thailand |
| Bang Tao Beach | Phuket, Thailand |
| Thalang District | Phuket, Thailand |
| Talat Nuea | Mueang Phuket District, Thailand |
| Kata Noi Beach | Karon, Thailand |
| Wichit | Wichit, Thailand |
| Nai Yang | Phuket, Thailand |
| Talat Yai | Mueang Phuket District, Thailand |
| Koh Kaew | Ko Kaeo, Thailand |
| Nai Thon | Phuket, Thailand |
| Ratsada | Phuket Island, Thailand |

For our study, we would be using sentiment analysis algorithm on 30 hotels/restaurants in the Kamala area of Thailand.

```
In [5]: dataset = reviews[reviews.Location == ' Kamala']
```

```
In [6]: dataset.describe()
```

Out[6]:

| | ID | Review Date | Location | Hotel/Restaurant name | Review |
|---|---|---|---|---|---|
| **count** | 3162 | 3162 | 3162 | 3162 | 3162 |
| **unique** | 2932 | 1070 | 1 | 32 | 2930 |
| **top** | rn579371699 | Reviewed 1 week ago | Kamala | HQ Beach Lounge | Perfect place to unwind. Food is very tasty an... |
| **freq** | 6 | 51 | 3162 | 100 | 6 |

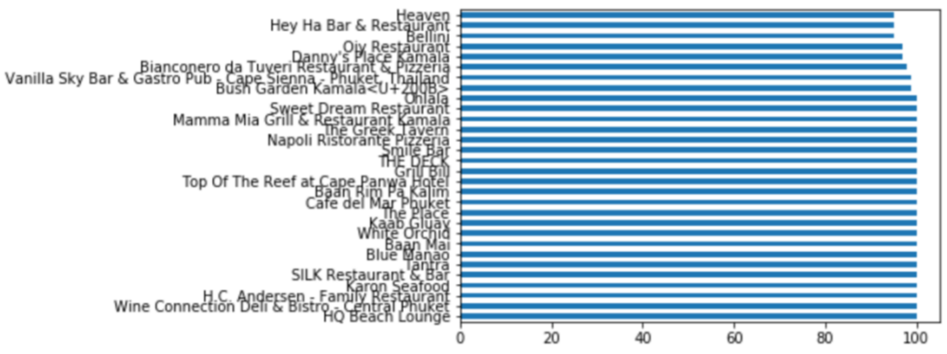
We can see that there are 32 hotels/restaurants in this area. So, we would be selecting 30 out of the 32. We did a value count for the 32 Hotels/restaurants and then we removed the two with the least values. The description for our final dataset is shown below.

```
In [12]: dataset.describe()
```

Out[12]:

| | ID | Review Date | Location | Hotel/Restaurant name | Review |
|---|---|---|---|---|---|
| **count** | 2975 | 2975 | 2975 | 2975 | 2975 |
| **unique** | 2795 | 1021 | 1 | 30 | 2793 |
| **top** | rn579633093 | Reviewed 1 week ago | Kamala | HQ Beach Lounge | The restaurant has wide terrace space to dinni... |
| **freq** | 6 | 48 | 2975 | 100 | 6 |

```
In [23]: #To see how many reviews each hotel has

         dataset['Hotel/Restaurant name'].value_counts()[:30].plot(kind='barh')

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9d6755d290>
```

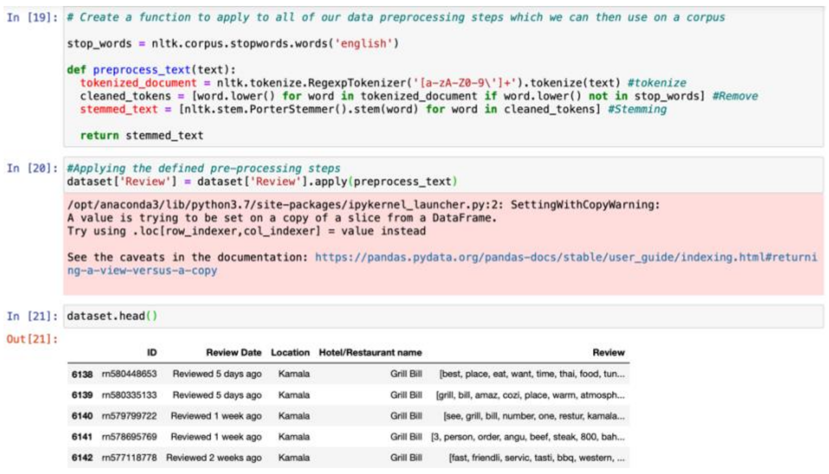


A **corpus** is a group of text documents that we would analyse using text mining or natural language processing techniques to draw conclusions from.

**Tokenization** is the process of breaking down a large block of text into smaller, more manageable pieces called **Tokens**.

A text normalisation technique called **stemmatization** changes every form of word to its base root mode.

The first step is to establish the corpus and then we tokenize, and then we clean by removing punctuations, stopwords, changing all to texts to lowercase after which we stemmatize. Here, we used Porter's stemmer.



```
In [19]: # Create a function to apply to all of our data preprocessing steps which we can then use on a corpus

         stop_words = nltk.corpus.stopwords.words('english')

         def preprocess_text(text):
           tokenized_document = nltk.tokenize.RegexpTokenizer('[a-zA-Z0-9\']+').tokenize(text) #tokenize
           cleaned_tokens = [word.lower() for word in tokenized_document if word.lower() not in stop_words] #Remove
           stemmed_text = [nltk.stem.PorterStemmer().stem(word) for word in cleaned_tokens] #Stemming

           return stemmed_text
```

```
In [20]: #Applying the defined pre-processing steps
         dataset['Review'] = dataset['Review'].apply(preprocess_text)

         /opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
         A value is trying to be set on a copy of a slice from a DataFrame.
         Try using .loc[row_indexer,col_indexer] = value instead

         See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
         ng-a-view-versus-a-copy
```

```
In [21]: dataset.head()
Out[21]:
```

| | ID | Review Date | Location | Hotel/Restaurant name | Review |
|---|---|---|---|---|---|
| **6138** | rn580448653 | Reviewed 5 days ago | Kamala | Grill Bill | [best, place, eat, want, time, thai, food, tun... |
| **6139** | rn580335133 | Reviewed 5 days ago | Kamala | Grill Bill | [grill, bill, amaz, cozi, place, warm, atmosph... |
| **6140** | rn579799722 | Reviewed 1 week ago | Kamala | Grill Bill | [see, grill, bill, number, one, restur, kamala... |
| **6141** | rn578695769 | Reviewed 1 week ago | Kamala | Grill Bill | [3, person, order, angu, beef, steak, 800, bah... |
| **6142** | rn577118778 | Reviewed 2 weeks ago | Kamala | Grill Bill | [fast, friendli, servic, tasti, bbq, western, ... |

## Implementation in Python

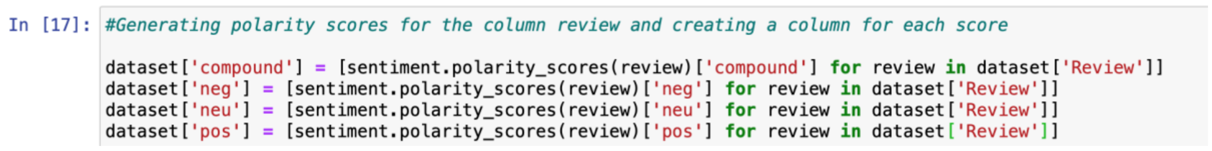
To implement sentiment analysis in python, we would be using the VADER (Valence Aware Dictionary for Sentiment Reasoning) model for the text sentiment analysis.

### VADER (Valence Aware Dictionary for Sentiment Reasoning)

**VADER** is a text sentiment analysis model that considers both the polarity (positive/negative) and strength (intensity) of the emotion. It uses a dictionary that converts lexical features into sentiment scores which is a measure of the intensity of an emotion. It tells us not only the sentiment scores but also about the degree of positive and negative emotions (Aditya Beri, 2020).

We chose to use this model because it doesn't require any training data, it is suitable for reviews and social media text, it is very good at detecting and understanding the sentiment of texts containing words, slangs, conjunctions, symbols and so much more.

### Creating the Model in Python



```
In [17]: #Generating polarity scores for the column review and creating a column for each score

         dataset['compound'] = [sentiment.polarity_scores(review)['compound'] for review in dataset['Review']]
         dataset['neg'] = [sentiment.polarity_scores(review)['neg'] for review in dataset['Review']]
         dataset['neu'] = [sentiment.polarity_scores(review)['neu'] for review in dataset['Review']]
         dataset['pos'] = [sentiment.polarity_scores(review)['pos'] for review in dataset['Review']]
```

```
In [18]: dataset.head()
```

Out[18]:

| | ID | Review Date | Location | Hotel/Restaurant name | Review | compound | neg | neu | pos |
|---|---|---|---|---|---|---|---|---|---|
| **6138** | rn580448653 | Reviewed 5 days ago | Kamala | Grill Bill | The best place to eat when you want time out f... | 0.9600 | 0.000 | 0.686 | 0.314 |
| **6139** | rn580335133 | Reviewed 5 days ago | Kamala | Grill Bill | Grill Bill is a amazing cozy place with a warm... | 0.9704 | 0.000 | 0.565 | 0.435 |
| **6140** | rn579799722 | Reviewed 1 week ago | Kamala | Grill Bill | After seeing that Grill Bill was the number on... | 0.9042 | 0.000 | 0.742 | 0.258 |
| **6141** | rn578695769 | Reviewed 1 week ago | Kamala | Grill Bill | 3 persons ordered Angus Beef steak 800 baht ea... | 0.1833 | 0.147 | 0.662 | 0.191 |
| **6142** | rn577118778 | Reviewed 2 weeks ago | Kamala | Grill Bill | Very fast and friendly service, tasty BBQ, but... | 0.3031 | 0.000 | 0.870 | 0.130 |

```
In [20]: #Getting a general overview of the scores
         dataset[['compound','neg','neu','pos']].describe()
```

Out[20]:

| | compound | neg | neu | pos |
|---|---|---|---|---|
| **count** | 2975.000000 | 2975.000000 | 2975.000000 | 2975.000000 |
| **mean** | 0.668452 | 0.027218 | 0.734395 | 0.238391 |
| **std** | 0.420118 | 0.050901 | 0.124043 | 0.133638 |
| **min** | -0.932800 | 0.000000 | 0.244000 | 0.000000 |
| **25%** | 0.612450 | 0.000000 | 0.657000 | 0.141000 |
| **50%** | 0.844200 | 0.000000 | 0.738000 | 0.231000 |
| **75%** | 0.929900 | 0.043000 | 0.822000 | 0.326000 |
| **max** | 0.990400 | 0.427000 | 1.000000 | 0.756000 |

The compound score is between -1 and 1. If it is greater than 0, it is a positive review and vice versa. The median here is 0.84 which means that most of the scores are above zero which means majority are positive reviews.

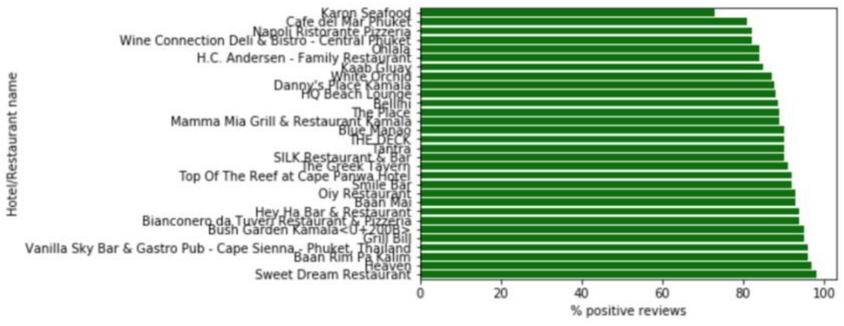
Now, we would look at the positive scores per restaurant/hotel just to get an overlook of the ratings for each.

```
In [22]: #calculate positive reviews as percentage of total reviews for each restaurant/hotel

         percent_positive = pd.DataFrame((dataset['compound']>0).groupby(dataset['Hotel/Restaurant name']).sum()
         /dataset['Hotel/Restaurant name'].groupby(dataset['Hotel/Restaurant name']).count()*100, columns = ['% positive revi

         percent_positive
```

Out[22]:

| | % positive reviews |
|---|---|
| **Hotel/Restaurant name** | |
| **Karon Seafood** | 73.000000 |
| **Cafe del Mar Phuket** | 81.000000 |
| **Napoli Ristorante Pizzeria** | 82.000000 |
| **Wine Connection Deli & Bistro - Central Phuket** | 82.000000 |
| **Ohlala** | 84.000000 |
| **H.C. Andersen - Family Restaurant** | 84.000000 |
| **Kaab Gluay** | 85.000000 |
| **White Orchid** | 87.000000 |
| **Danny's Place Kamala** | 87.628866 |
| **HQ Beach Lounge** | 88.000000 |

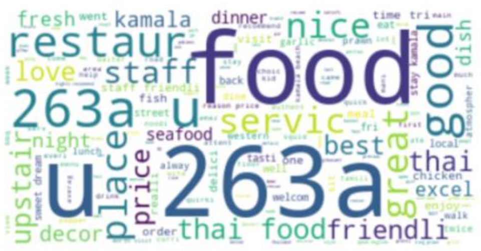
```
In [23]: sns.barplot(data=percent_positive, x = '% positive reviews', y=percent_positive.index, color='green')

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7fcc1d01f050>
```



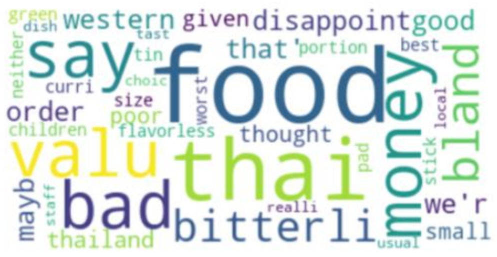
We can see that ‘Karon seafood’ has the lowest positive review and ‘Sweet Dream Restaurant’ has the highest positive review.

Now we will build a word cloud of positive and negative reviews for the restaurant with the highest positive review.

```
In [27]: #wordcloud of words from positive reviews for Sweet Dream restaurant

         pos_tokens = [word for review in dataset_positive_subset['processed_review'] for word in review]

         wordcloud = WordCloud(background_color='white').generate_from_text( ' '.join(pos_tokens))

         #Display the generated image:
         plt.figure(figsize=(8,8))
         plt.imshow(wordcloud, interpolation='bilinear')
         plt.axis("off")
         plt.show

Out[27]: <function matplotlib.pyplot.show(*args, **kw)>
```



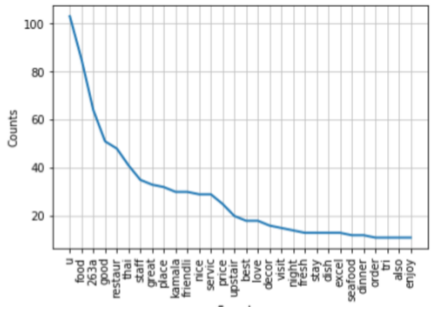
```
In [28]: #wordcloud of words from positive reviews for Sweet Dream restaurant

         neg_tokens = [word for review in dataset_negative_subset['processed_review'] for word in review]

         wordcloud = WordCloud(background_color='white').generate_from_text( ' '.join(neg_tokens))

         #Display the generated image:
         plt.figure(figsize=(8,8))
         plt.imshow(wordcloud, interpolation='bilinear')
         plt.axis("off")
         plt.show

Out[28]: <function matplotlib.pyplot.show(*args, **kw)>
```



To view the word counts to see which words were the most frequent for both the positive and negative reviews, we used the Frequency Distribution library to plot this.

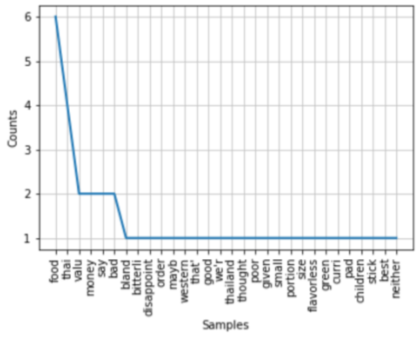
```
In [33]: #use the nltk FreqDist to view word count, tabulate then visualize
         from nltk.probability import FreqDist
         pos_freqdist = FreqDist(pos_tokens)
         pos_freqdist.tabulate(10)
         pos_freqdist.plot(30)
```

```
      u     food    263a    good restaur    thai   staff   great   place  kamala
    103      85      64      51      48      41      35      33      32      30
```



```
In [34]: #use the nltk FreqDist to view word count, tabulate then visualize
         neg_freqdist = FreqDist(neg_tokens)
         neg_freqdist.tabulate(10)
         neg_freqdist.plot(30)
```

```
     food         thai         valu        money          say          bad        bland   bitterli d
        6            4            2            2            2            2            1            1
```



## Results analysis and discussion

We realised that for the implementation of VADER, we didn't need to pre-process the data. When we implemented the algorithm with the pre-processed data, we got the following error « 'list' object has no attribute 'encode' », so we had to use the raw reviews in strings for the modelling. And to build the wordcloud to visually see the frequent and distribution of words, we needed to tokenize, remove stopwords and stem the text to be able to have a word count. So, the pre-processing algorithm was implemented here.

For the sweet dream restaurant, we can clearly see the main words that contribute to the positive and negative reviews.

## Conclusions

VADER is a very efficient model for sentiment analysis. It is very easy to use, and the results are concise, clear, and compelling. The study above helps the restaurants/hotels in the Kamala area of Thailand see and understand their strong and weak links which in turn can help them increase their quality of service in the right areas which will then lead to improved positive scores.