In [ ]:
```
import nltk
nltk.download('wordnet')
import numpy as np
import pandas as pd
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.wordnet import WordNetLemmatizer
from ast import literal_eval
```

In [2]:
```
import os
```

In [11]:
```
import nltk
nltk.download('wordnet')
import numpy as np
import pandas as pd
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.wordnet import WordNetLemmatizer
from ast import literal_eval
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\chinmayee\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

In [ ]:

In [ ]:

In [12]:
```
os.getcwd()
```

Out[12]:
```
'C:\\Users\\chinmayee'
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [13]:
```
data = pd.read_csv(r"C:\Users\chinmayee\Downloads\hotel\hotels.csv")
data.head()
```

Out[13]:

	hotel_address	additional_number_of_scoring	review_date	average_score	hotel_name	revie
0	Stratton Street Mayfair Westminster Borough Lo...	581	2/19/2016	8.4	The May Fair Hotel	
1	130 134 Southampton Row Camden London WC1B 5AF...	299	1/12/2017	8.3	Mercure London Bloomsbury Hotel	
2	151 bis Rue de Rennes 6th arr 75006 Paris France	32	10/18/2016	8.9	Legend Saint Germain by Elegancia	
3	216 Avenue Jean Jaures 19th arr 75019 Paris Fr...	34	9/22/2015	7.5	Mercure Paris 19 Philharmonie La Villette	
4	Molenwerf 1 1014 AG Amsterdam Netherlands	914	3/5/2016	8.5	Golden Tulip Amsterdam West	

In [ ]:

In [14]:
```
data = pd.read_csv(r"C:\Users\chinmayee\Downloads\hotel\Hotel_Reviews.csv")
data.head()
```

Out[14]:

	Hotel_Address	Additional_Number_of_Scoring	Review_Date	Average_Score	Hotel_Name
0	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	194	8/3/2017	7.7	Hotel Arena
1	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	194	8/3/2017	7.7	Hotel Arena
2	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	194	7/31/2017	7.7	Hotel Arena
3	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	194	7/31/2017	7.7	Hotel Arena
4	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	194	7/24/2017	7.7	Hotel Arena

In [15]:
```
# Replacing "United Kingdom with "UK"
data.Hotel_Address = data.Hotel_Address.str.replace("United Kingdom", "UK")
```

In [16]:
```
# Now I will split the address and pick the last word in the address to identify
data["countries"] = data.Hotel_Address.apply(lambda x: x.split(' ')[-1])
```

In [17]:
```
print(data.countries.unique())
```

```
['Netherlands' 'UK' 'France' 'Spain' 'Italy' 'Austria']
```

In [18]:
```
data.drop(['Additional_Number_of_Scoring',
       'Review_Date','Reviewer_Nationality',
       'Negative_Review', 'Review_Total_Negative_Word_Counts',
       'Total_Number_of_Reviews', 'Positive_Review',
       'Review_Total_Positive_Word_Counts',
       'Total_Number_of_Reviews_Reviewer_Has_Given', 'Reviewer_Score',
       'days_since_review', 'lat', 'lng'],1,inplace=True)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[18], line 1
----> 1 data.drop(['Additional_Number_of_Scoring',
      2        'Review_Date','Reviewer_Nationality',
      3        'Negative_Review', 'Review_Total_Negative_Word_Counts',
      4        'Total_Number_of_Reviews', 'Positive_Review',
      5        'Review_Total_Positive_Word_Counts',
      6        'Total_Number_of_Reviews_Reviewer_Has_Given', 'Reviewer_Score',
      7        'days_since_review', 'lat', 'lng'],1,inplace=True)

TypeError: DataFrame.drop() takes from 1 to 2 positional arguments but 3 positi
onal arguments (and 1 keyword-only argument) were given
```

In [19]:
```
# Drop multiple columns from the DataFrame
columns_to_drop = [
    'Additional_Number_of_Scoring',
    'Review_Date',
    'Reviewer_Nationality',
    'Negative_Review',
    'Review_Total_Negative_Word_Counts',
    'Total_Number_of_Reviews',
    'Positive_Review',
    'Review_Total_Positive_Word_Counts',
    'Total_Number_of_Reviews_Reviewer_Has_Given',
    'Reviewer_Score',
    'days_since_review',
    'lat',
    'lng'
]

data.drop(columns_to_drop, axis=1, inplace=True)
```

In [20]:
```
def impute(column):
    column = column[0]
    if (type(column) != list):
        return "".join(literal_eval(column))
    else:
        return column

data["Tags"] = data[["Tags"]].apply(impute, axis=1)
data.head()
```

Out[20]:

	Hotel_Address	Average_Score	Hotel_Name	Tags	countries
0	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	7.7	Hotel Arena	Leisure trip Couple Duplex Double Room Sta...	Netherlands
1	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	7.7	Hotel Arena	Leisure trip Couple Duplex Double Room Sta...	Netherlands
2	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	7.7	Hotel Arena	Leisure trip Family with young children Dup...	Netherlands
3	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	7.7	Hotel Arena	Leisure trip Solo traveler Duplex Double Ro...	Netherlands
4	s Gravesandestraat 55 Oost 1092 AA Amsterdam ...	7.7	Hotel Arena	Leisure trip Couple Suite Stayed 2 nights ...	Netherlands

In [21]:
```
data['countries'] = data['countries'].str.lower()
data['Tags'] = data['Tags'].str.lower()
```

In [22]:
```
def recommend_hotel(location, description):
    description = description.lower()
    word_tokenize(description)
    stop_words = stopwords.words('english')
    lemm = WordNetLemmatizer()
    filtered  = {word for word in description if not word in stop_words}
    filtered_set = set()
    for fs in filtered:
        filtered_set.add(lemm.lemmatize(fs))

    country = data[data['countries']==location.lower()]
    country = country.set_index(np.arange(country.shape[0]))
    list1 = []; list2 = []; cos = [];
    for i in range(country.shape[0]):
        temp_token = word_tokenize(country["Tags"][i])
        temp_set = [word for word in temp_token if not word in stop_words]
        temp2_set = set()
        for s in temp_set:
            temp2_set.add(lemm.lemmatize(s))
        vector = temp2_set.intersection(filtered_set)
        cos.append(len(vector))
    country['similarity']=cos
    country = country.sort_values(by='similarity', ascending=False)
    country.drop_duplicates(subset='Hotel_Name', keep='first', inplace=True)
    country.sort_values('Average_Score', ascending=False, inplace=True)
    country.reset_index(inplace=True)
    return country[["Hotel_Name", "Average_Score", "Hotel_Address"]].head()
```

In [23]:
```
recommend_hotel('Italy', 'I am going for a business trip')
```

Out[23]:

	Hotel_Name	Average_Score	Hotel_Address
0	Excelsior Hotel Gallia Luxury Collection Hotel	9.4	Piazza Duca D Aosta 9 Central Station 20124 Mi...
1	Palazzo Parigi Hotel Grand Spa Milano	9.3	Corso Di Porta Nuova 1 Milan City Center 20121...
2	Hotel Spadari Al Duomo	9.3	Via Spadari 11 Milan City Center 20123 Milan I...
3	Room Mate Giulia	9.3	Silvio Pellico 4 Milan City Center 20121 Milan...
4	UNA Maison Milano	9.3	Via Mazzini 4 Milan City Center 20123 Milan Italy

In [ ]:
```
recommend_hotel('spain', 'I am going for a holiday')
```

In [ ]:

In [ ]: