Indian Institute of Management
Ahmedabad

e-Post Graduate Diploma in Advanced Business Analytics, 2021-2022

# Big Data Analytics - Analysis of Text and Social Media Data

## Assignment 2

Bharath Dasari
PGDAE202100006

# Assignment 2: Topic Modelling

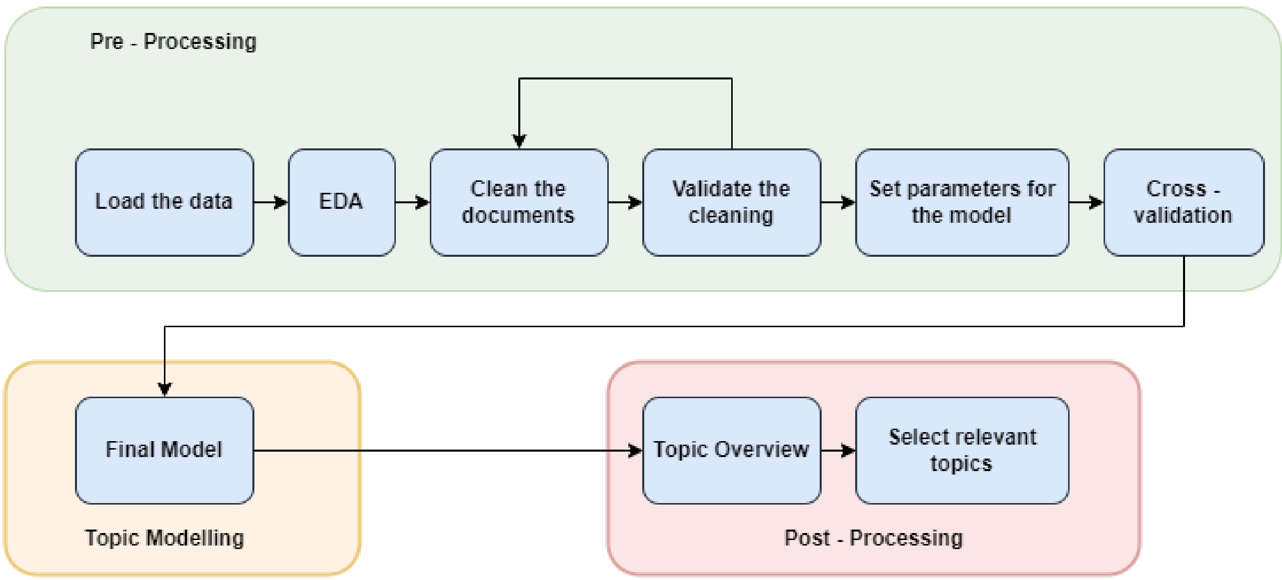
## Introduction:

Topic modelling is an unsupervised machine learning technique that's capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize a set of documents. By using topic analysis models, businesses can offload simple tasks onto machines instead of overloading employees with too much data. Just imagine the time your team could save and spend on more important tasks if a machine was able to sort through endless lists of customer reviews or support tickets.

## Objective:

The primary objective is to analyse the presented data assuming the position of a Brand Manager of a particular firm and derive meaningful insights to make an informed decision about how the firm is faring compared to its competitors and for a better brand positioning.

## Methodology:




## Datasets:

The data sets used for this analysis are the files which contain reviews of three different hotels namely – Oberoi, Park and Radisson from the city of Mumbai. Each of these files consists of approximately 1000 reviews.

**Features:**

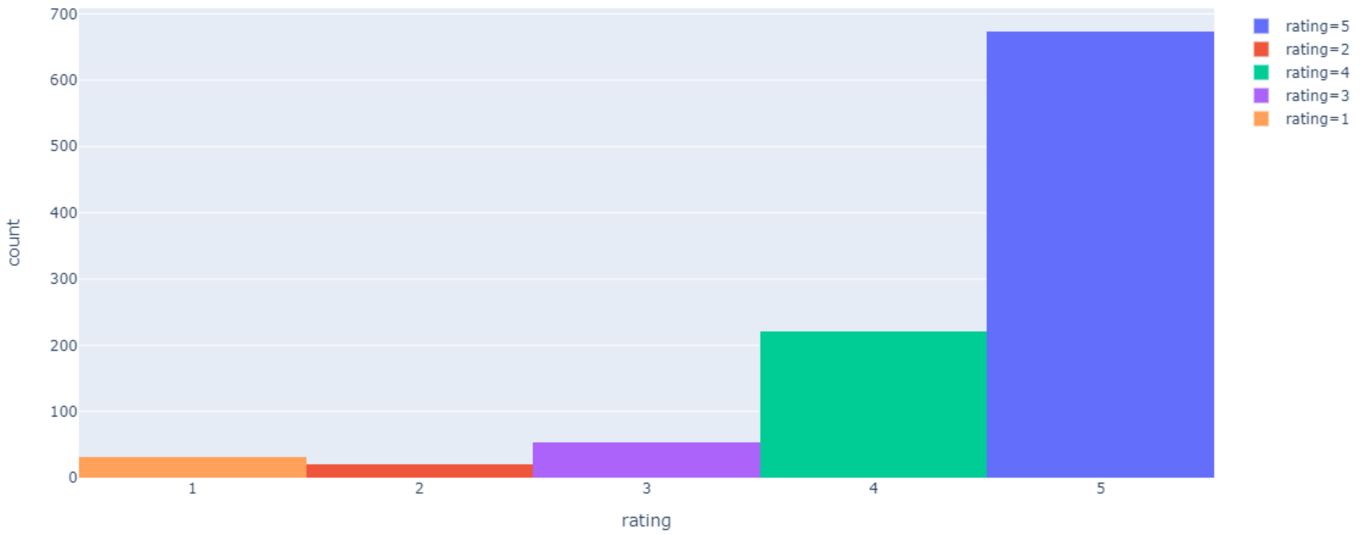
Review title**,** Date, Reviewer, Rating, Review

We shall be using the Date, Rating and Reviews for further exploratory data analysis and the review data is used for modelling after appropriate text pre-processing.

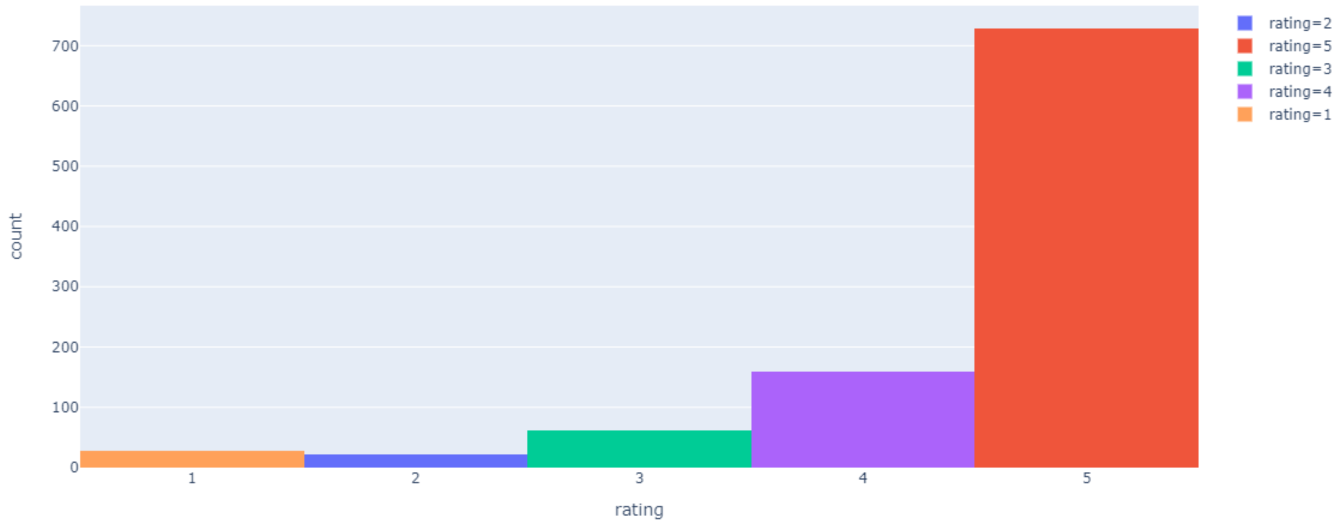
## EDA:

After looking at the EDA report, there are very few duplicates (<1%).

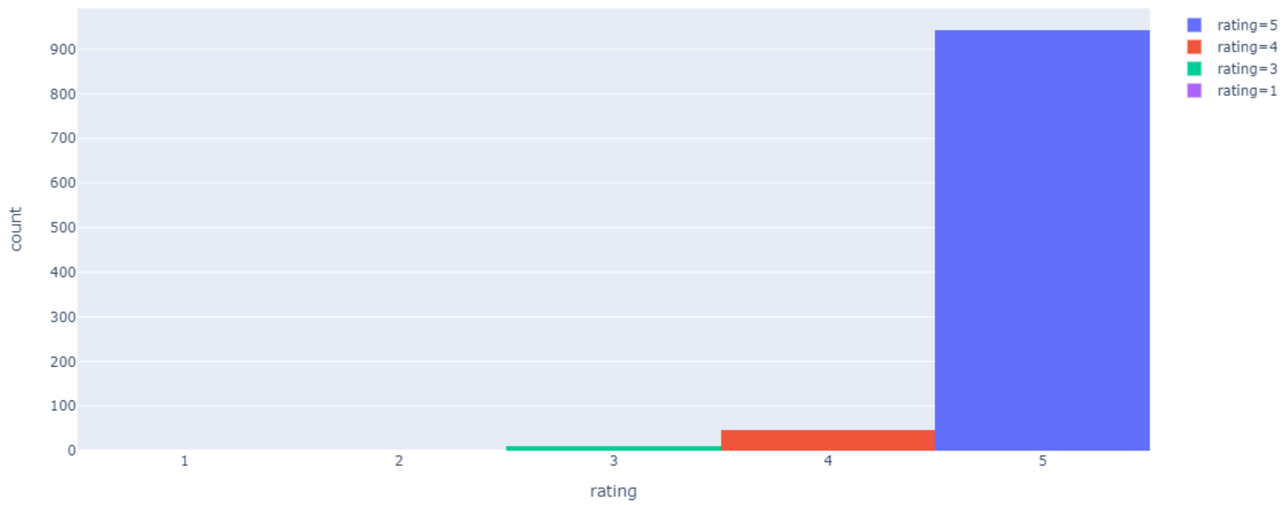
**Radisson rating distribution:**



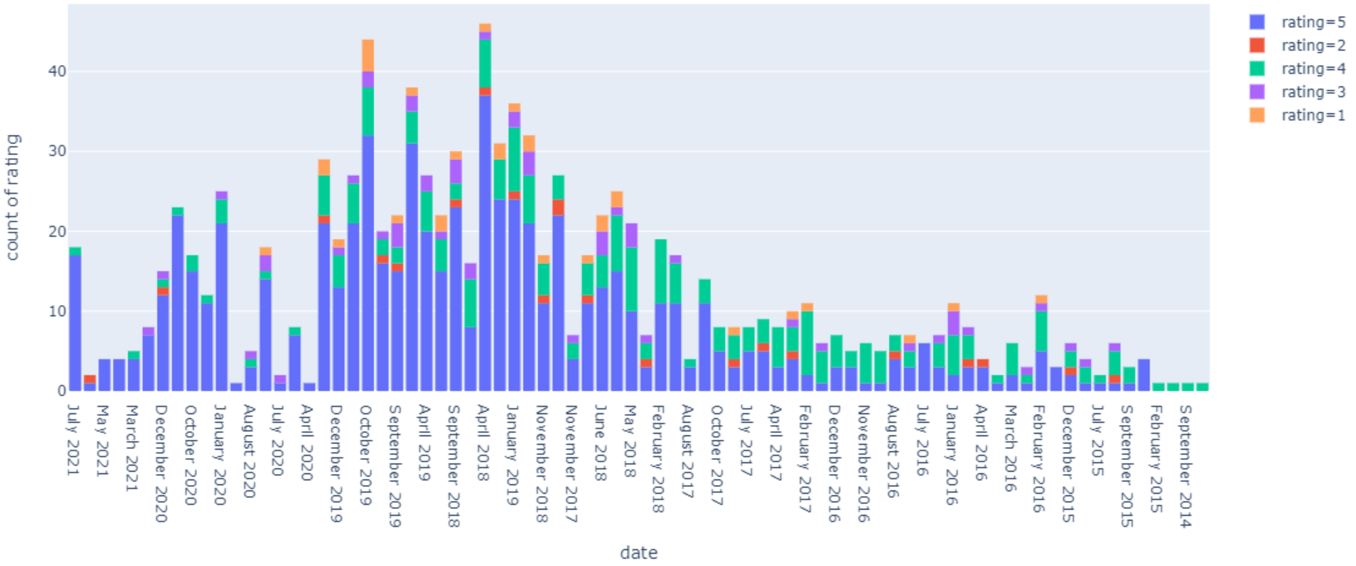

**Taj rating distribution:**



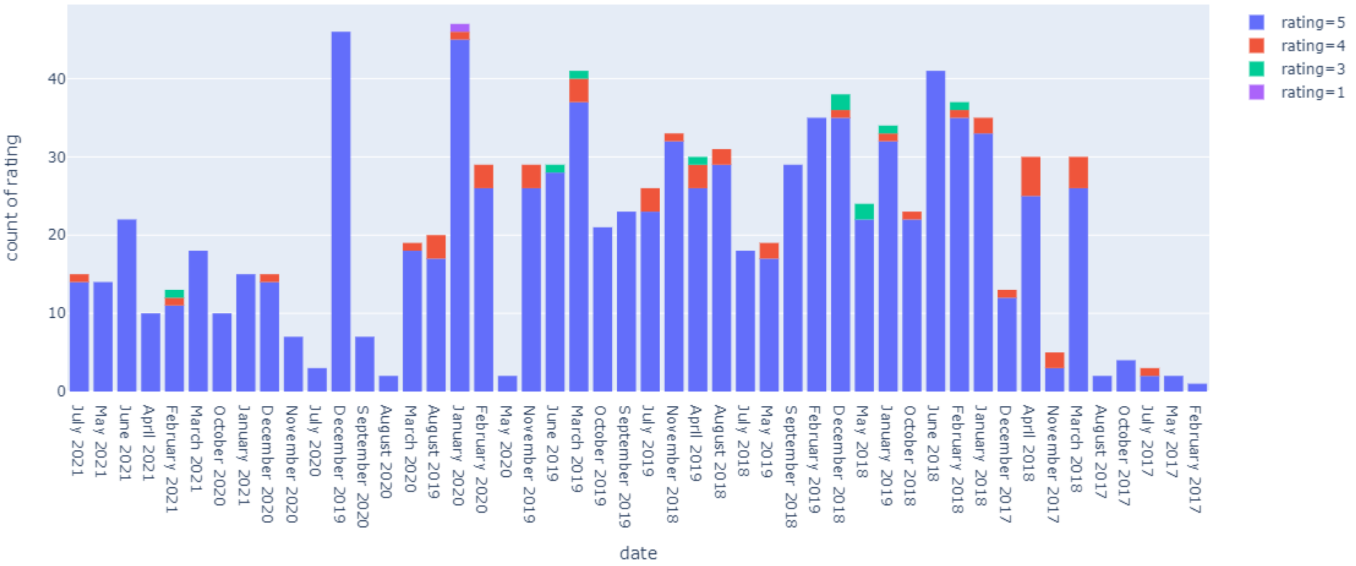

**Oberoi rating distribution:**



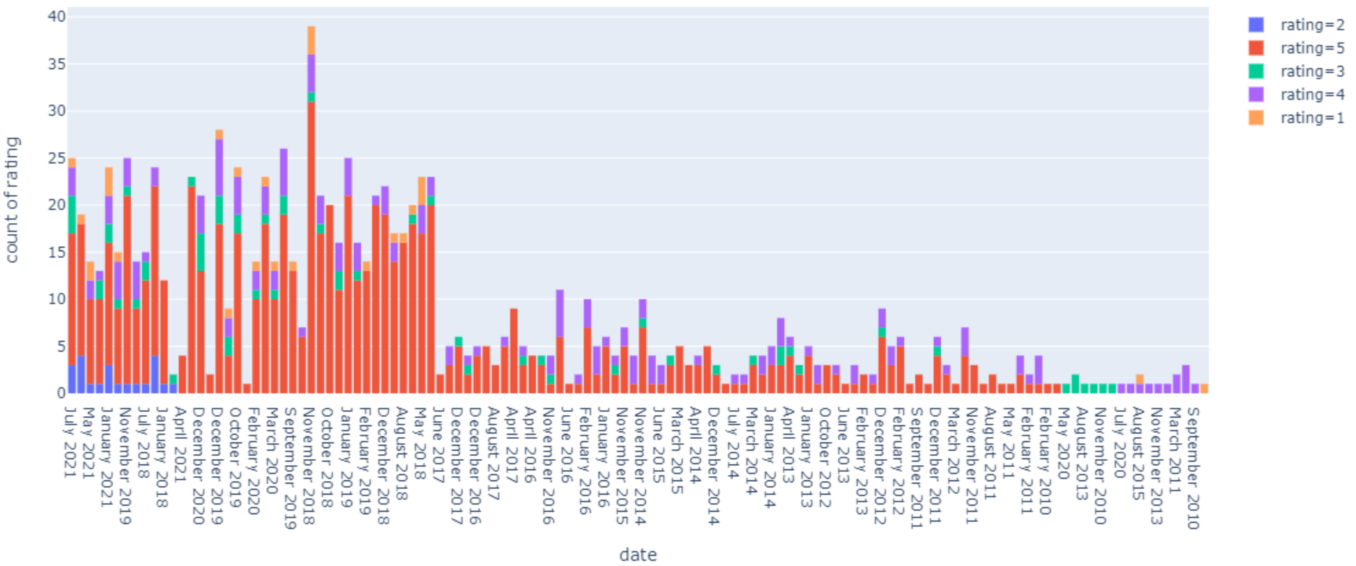
## Radisson hotel




## Oberoi hotel




## Taj hotel:



From the above distributions, we can clearly observe that, all the three hotels are performing exceptionally good in terms of the reviews they got over a period of 7 – 8 years on an average as most of their rating are on the higher side. This tells us that, if a brand manager must infer some insights out of this, he can certainly look at the time periods for which each of the hotel gets more bookings and what are the rating distributions for those bookings and compare them against each other for the better understating od where a certain hotel can improve on. For example, Radisson and Taj hotel have got almost equal distribution of ratings but then, the time periods for those differ and if we further drill down, Taj hotel got most of those ratings starting from the second quarter of 2018 with majority of them being 5 and 4. Now, if we compare those ratings against Radisson, we can see that the ratings are same, but the number of different ratings differ. If we want to understand it as what makes the customers better like either of these hotels, we can perform the topic modelling which gives us what are the distinct features that pulls the customers towards a particular hotel and what are those features that the customers like about their stay at the hotel. This information comes in handy while a brand manager performs competitive analysis.

## Pre-Processing:

In order to get fair understanding of the review data and what are the words that stand out in most of the reviews, we try and build the word clouds for the review data and see the most used words. Before going to this step, the data has been pre-processed which involves text cleansing, tokenization, stop words removal, lemmatization and stemming.

**Word clouds:**


*Taj*


*Radisson*


*Oberoi*

From these word clouds, we can see what are the top spoken topics in the reviews which helps us in the further analysis of what can be the top topics which can be used for decision making.

## Topic Modelling:

I have tried using three models for this topic modelling of hotel reviews.

### LSI Model:

Radisson:

```
from gensim import corpora, models
lsi_model = models.LsiModel(corpus=corpus, num_topics=10, id2word=id2word)
lsi_model.show_topics(num_topics=5)
```

```
[(0,
  '0.382*"good" + 0.344*"stay" + 0.341*"hotel" + 0.320*"staff" + 0.294*"room" + 0.257*"number" + 0.225*"service" + 0.193*"food" + 0.114*"g
 (1,
  '0.689*"good" + -0.436*"number" + -0.297*"hotel" + -0.222*"stay" + 0.197*"service" + -0.193*"room" + 0.162*"food" + -0.087*"day" + 0.072
 (2,
  '0.463*"room" + 0.397*"good" + 0.289*"hotel" + -0.287*"stay" + -0.267*"staff" + -0.229*"thank" + -0.201*"great" + -0.195*"service" + -0.
 (3,
  '0.711*"number" + -0.412*"stay" + -0.242*"hotel" + 0.239*"food" + 0.207*"service" + -0.145*"staff" + -0.140*"room" + -0.093*"helpful" +
 (4,
  '0.688*"hotel" + -0.483*"stay" + -0.366*"room" + 0.191*"staff" + 0.117*"great" + -0.114*"number" + -0.113*"good" + 0.098*"food" + -0.061
```

```
# Compute Coherence Score
coherence_model_lsi = CoherenceModel(model=lsi_model, texts=data_lemmatized, dictionary=id2word, coherence='c_v')
coherence_lsi = coherence_model_lsi.get_coherence()
print('Coherence Score: ', coherence_lsi)
```

```
Coherence Score:  0.3659415600748722
```

Oberoi:

```
from gensim import corpora, models
lsi_model = models.LsiModel(corpus=corpus, num_topics=10, id2word=id2word)
lsi_model.show_topics(num_topics=5)
```

```
[(0,
  '0.440*"hotel" + 0.386*"stay" + 0.314*"room" + 0.252*"staff" + 0.228*"number" + 0.223*"service" + 0.196*"oberoi" + 0.186*"good" + 0.150
 (1,
  '-0.533*"stay" + 0.513*"room" + 0.393*"hotel" + -0.229*"make" + -0.225*"oberoi" + -0.183*"staff" + -0.116*"thank" + 0.107*"good" + -0.0
 (2,
  '-0.680*"hotel" + 0.558*"room" + 0.269*"number" + -0.170*"good" + 0.130*"check" + 0.118*"view" + 0.068*"night" + -0.058*"experience" + (
 (3,
  '-0.738*"number" + 0.327*"good" + -0.255*"hotel" + 0.226*"staff" + 0.154*"service" + 0.142*"great" + 0.127*"view" + 0.122*"room" + 0.09
 (4,
  '0.703*"good" + -0.356*"staff" + 0.283*"number" + -0.214*"make" + -0.191*"hotel" + 0.186*"service" + 0.157*"stay" + -0.131*"feel" + -0.
```

```
# Compute Coherence Score
coherence_model_lsi = CoherenceModel(model=lsi_model, texts=data_lemmatized, dictionary=id2word, coherence='c_v')
coherence_lsi = coherence_model_lsi.get_coherence()
print('Coherence Score: ', coherence_lsi)
```

```
Coherence Score:  0.37138711660024637
```

Taj:

```
from gensim import corpora, models
lsi_model = models.LsiModel(corpus=corpus, num_topics=10, id2word=id2word)
lsi_model.show_topics(num_topics=5)
```

```
[(0,
  '0.480*"room" + 0.409*"hotel" + 0.363*"stay" + 0.267*"number" + 0.188*"staff" + 0.162*"good" + 0.160*"service" + 0.150*"view" + 0.137*"t
 (1,
  '0.560*"number" + 0.507*"room" + -0.473*"hotel" + -0.258*"stay" + -0.117*"staff" + -0.101*"good" + -0.089*"great" + -0.079*"excellent" +
 (2,
  '-0.721*"number" + 0.566*"room" + -0.179*"hotel" + -0.151*"stay" + 0.113*"view" + -0.069*"experience" + -0.066*"make" + 0.060*"service"
 (3,
  '-0.677*"hotel" + 0.487*"stay" + 0.197*"experience" + 0.177*"staff" + 0.151*"make" + 0.144*"taj" + -0.144*"room" + 0.114*"good" + 0.098*
 (4,
  '0.616*"stay" + -0.407*"good" + -0.267*"staff" + -0.237*"service" + -0.227*"experience" + -0.177*"food" + -0.175*"great" + 0.153*"tower"
```

```
# Compute Coherence Score
coherence_model_lsi = CoherenceModel(model=lsi_model, texts=data_lemmatized, dictionary=id2word, coherence='c_v')
coherence_lsi = coherence_model_lsi.get_coherence()
print('Coherence Score: ', coherence_lsi)
```

```
Coherence Score:  0.4362060129230074
```

The above pictures shows the top 5 topics for each of the hotels.

**LDA Model:**

For the LDA model,  have built the model with arbitrary parameter values and then used hyper parameter tuning for obtaining the optimized values of no of topics, alpha and eta.

```
sorted_df = best_parms.sort_values(["Coherence"],
                                   ascending=False)
print(sorted_df)

    Validation_Set  Topics       Alpha  Beta  Coherence
253     75% Corpus      10        0.61  0.91   0.682134
208     75% Corpus       8  asymmetric  0.91   0.657332
238     75% Corpus       9  asymmetric  0.91   0.639549
123     75% Corpus       6        0.01  0.91   0.631265
178     75% Corpus       7  asymmetric  0.91   0.631095
..             ...     ...         ...   ...        ...
427    100% Corpus       7        0.31  0.61   0.315374
463    100% Corpus       8        0.61  0.91   0.311882
68      75% Corpus       4        0.31  0.91   0.297579
362    100% Corpus       5        0.01  0.61   0.297402
363    100% Corpus       5        0.01  0.91   0.291270
```

*Radisson*

```
sorted_df = best_parms.sort_values(["Coherence"],
                                   ascending=False)
print(sorted_df)

    Validation_Set  Topics      Alpha  Beta  Coherence
198     75% Corpus       8       0.91  0.91   0.674792
253     75% Corpus      10       0.61  0.91   0.665388
528    100% Corpus      10       0.91  0.91   0.660014
257     75% Corpus      10       0.91  0.61   0.654975
222     75% Corpus       9       0.61  0.61   0.635682
..             ...     ...        ...   ...        ...
91      75% Corpus       5       0.01  0.31   0.302623
111     75% Corpus       5  symmetric  0.31   0.297021
463    100% Corpus       8       0.61  0.91   0.296677
107     75% Corpus       5       0.91  0.61   0.281895
61      75% Corpus       4       0.01  0.31   0.275104
```

*Oberoi*

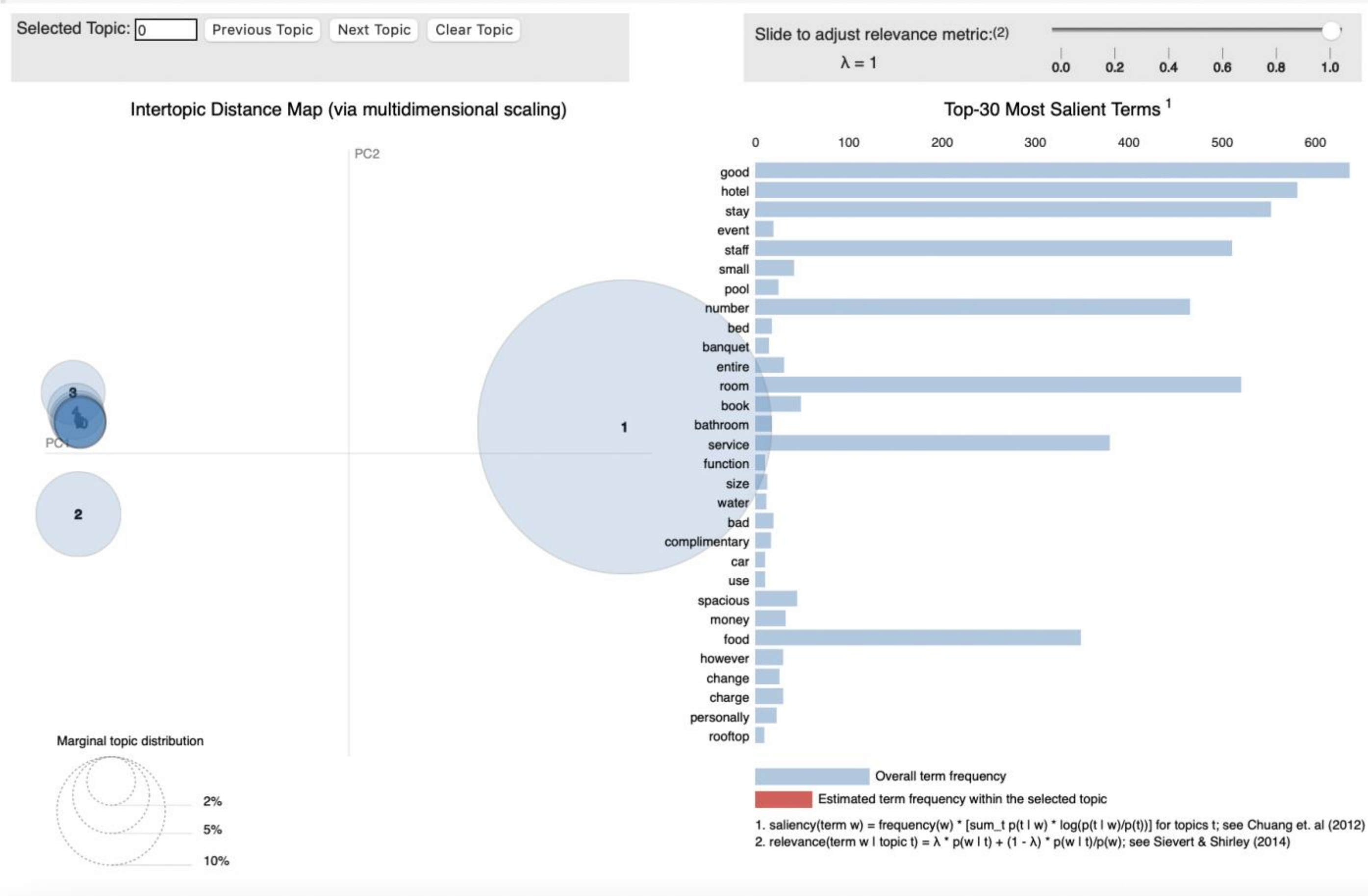
```
sorted_df = best_parms.sort_values(["Coherence"],
                                   ascending=False)
print(sorted_df)

    Validation_Set  Topics      Alpha       Beta  Coherence
263     75% Corpus      10  symmetric       0.91   0.594729
168     75% Corpus       7       0.91       0.91   0.583888
243     75% Corpus      10       0.01       0.91   0.578585
513    100% Corpus      10       0.01       0.91   0.577339
223     75% Corpus       9       0.61       0.91   0.569882
..             ...     ...        ...        ...        ...
31      75% Corpus       3       0.01       0.31   0.309872
34      75% Corpus       3       0.01  symmetric   0.309580
103     75% Corpus       5       0.61       0.91   0.307638
133     75% Corpus       6       0.61       0.91   0.305078
18      75% Corpus       2       0.91       0.91   0.303953
```

*Taj*

From the above analysis, we can see that the no of topics for each of the hotel varies and the key words which make up the topics differs as well.

Now, we try to visualize the topics for each of the hotels.

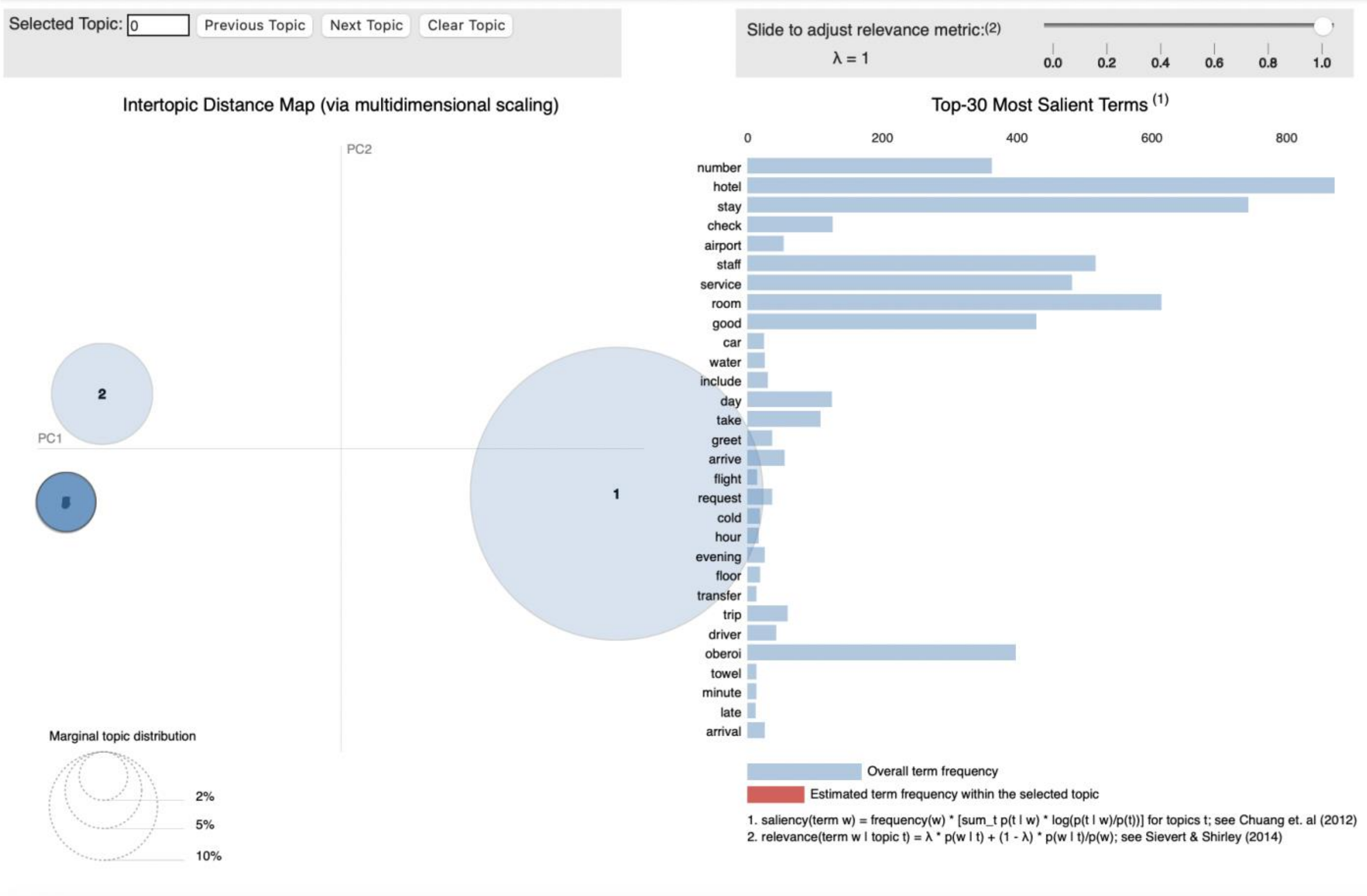
**Radisson:**




Each bubble on the left-hand side plot represents a topic. The larger the bubble, the more prevalent is that topic. A model with too many topics, will typically have many overlaps, small sized bubbles clustered in one region of the chart. If we move the cursor over one of the bubbles, the words and bars on the right-hand side will update. These words are the salient keywords that form the selected topic.

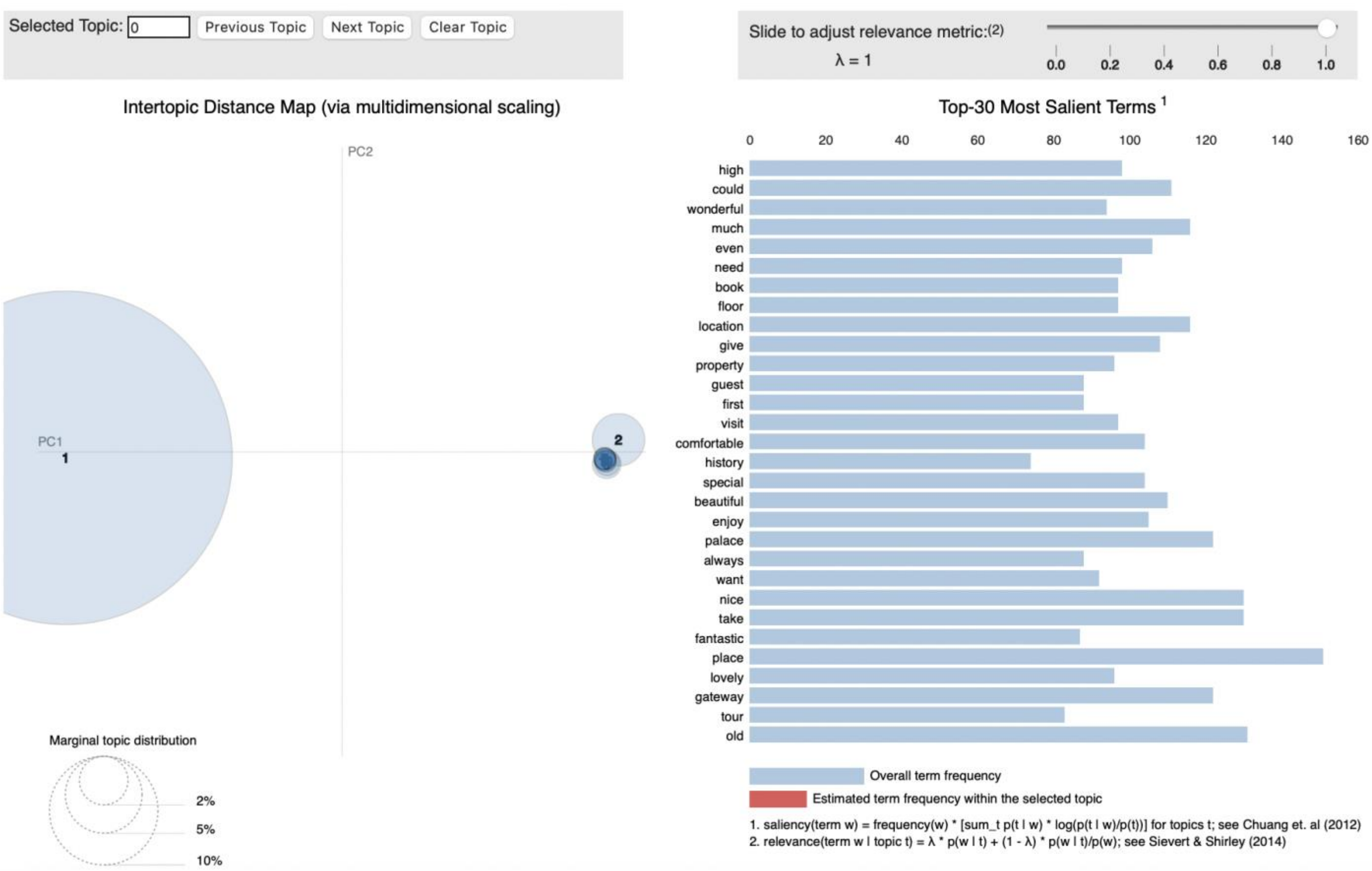
The top words for the Radisson hotel are good, hotel, stay, staff which characterizes the type of services which are most liked by the customers who have visited the hotel.

**Oberoi:**



Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0 0.2 0.4 0.6 0.8 1.0
Intertopic Distance Map (via multidimensional scaling)
Top-30 Most Salient Terms (1)
PC2
PC1
1
2
0 200 400 600 800
number
hotel
stay
check
airport
staff
service
room
good
car
water
include
day
take
greet
arrive
flight
request
cold
hour
evening
floor
transfer
trip
driver
oberoi
towel
minute
late
arrival
Marginal topic distribution
2%
5%
10%
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)


**Taj:**



Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0 0.2 0.4 0.6 0.8 1.0
Intertopic Distance Map (via multidimensional scaling)
Top-30 Most Salient Terms 1
PC2
PC1
1
2
0 20 40 60 80 100 120 140 160
high
could
wonderful
much
even
need
book
floor
location
give
property
guest
first
visit
comfortable
history
special
beautiful
enjoy
palace
always
want
nice
take
fantastic
place
lovely
gateway
tour
old
Marginal topic distribution
2%
5%
10%
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

We can see that the Taj hotel has good number of features which customers have liked during their stay at the hotel. One can try to do a competitive analysis of how they can try and add in those features if those are viable and profitable.

**Evaluation metrics used: Coherence Measure**

*Coherence: A set of statements or facts is said to be coherent, if they support each other. Thus, a coherent fact set can be interpreted in a context that covers all or most of the facts. An example of a coherent fact set is "the game is a team sport", "the game is played with a ball", "the game demands great physical efforts"*

***C_v** measure is based on a sliding window, one-set segmentation of the top words and an indirect confirmation measure that uses normalized pointwise mutual information (NPMI) and the cosine similarity*

**The limitations of LDA:**

Long documents are poorly represented because they have poor similarity values, search keywords must precisely match document terms; word substrings might result in a "false positive match", semantic sensitivity; documents with similar context but different term vocabulary won't be associated, resulting in a "false negative match", the order in which the terms appear in the document is lost in the vector space representation, theoretically assumes terms are statistically independent and last weighting is intuitive but not very formal.

**LDA Mallet model:**

```
ldamallet = gensim.models.wrappers.LdaMallet(mallet_path, corpus=corpus, num_topics=10, id2word=id2word)
```

```
# Show Topics
pprint(ldamallet.show_topics(formatted=False))

# Compute Coherence Score
coherence_model_ldamallet = CoherenceModel(model=ldamallet, texts=data_lemmatized, dictionary=id2word, coherence='c_v')
coherence_ldamallet = coherence_model_ldamallet.get_coherence()
print('\nCoherence Score: ', coherence_ldamallet)
```

```
[(0,
  [('view', 0.12313895781637717),
   ('stay', 0.07847394540942929),
   ('night', 0.05893300248138958),
   ('room', 0.05676178660049628),
   ('wing', 0.056451612903225805),
   ('tower', 0.04652605459057072),
   ('gateway', 0.04218362282878412),
   ('palace', 0.033498759305210915),
   ('floor', 0.028846153846153848),
   ('upgrade', 0.024193548387096774)]),
 (1
```

I have used LDA mallet model for three hotels data. Below are the results of the analysis.

```
Num Topics = 2  has Coherence Value of 0.3573
Num Topics = 8  has Coherence Value of 0.3922
Num Topics = 14  has Coherence Value of 0.3703
Num Topics = 20  has Coherence Value of 0.3467
Num Topics = 26  has Coherence Value of 0.3725
Num Topics = 32  has Coherence Value of 0.3897
Num Topics = 38  has Coherence Value of 0.4305
```

*Radisson*

```
Num Topics = 2  has Coherence Value of 0.3361
Num Topics = 8  has Coherence Value of 0.3898
Num Topics = 14  has Coherence Value of 0.3579
Num Topics = 20  has Coherence Value of 0.3455
Num Topics = 26  has Coherence Value of 0.3639
Num Topics = 32  has Coherence Value of 0.3695
Num Topics = 38  has Coherence Value of 0.3777
```

*Oberoi*

```
Num Topics = 2  has Coherence Value of 0.3878
Num Topics = 8  has Coherence Value of 0.4177
Num Topics = 14  has Coherence Value of 0.3672
Num Topics = 20  has Coherence Value of 0.337
Num Topics = 26  has Coherence Value of 0.3359
Num Topics = 32  has Coherence Value of 0.3477
Num Topics = 38  has Coherence Value of 0.3545
```

*Taj*

LDA assumes documents are produced from a mixture of topics. Those topics then generate words based on their probability distribution. Gensim is a python library with good tools to work on topic modelling, however genism does not provide an out of the box running commands to perform topic modelling, it requires python knowledge. Mallet is an out of the box tool but unfortunately it doesn't let you tweak or see in between steps like GenSim does. Mallet regardless being a console application is much more user friendly than GenSim, but for advanced work is better to use GenSim as it lets you tweak more parameters than Mallet.

## Applications:

### The dominant topic in each sentence:

One of the practical application of topic modelling is to determine what topic a given document is about. To find that, we find the topic number that has the highest percentage contribution in that document.

| Document_No | Dominant_Topic | Topic_Perc_Contrib | Keywords | Text |
|---|---|---|---|---|
| 0 | 2.0 | 0.2486 | number, check, time, book, room, upgrade, thin... | taj should relook and compare yourself with y... |
| 1 | 0.0 | 0.1617 | experience, taj, feel, amazing, property, love... | every time i stay here i feel welcomes and lo... |
| 2 | 0.0 | 0.2836 | experience, taj, feel, amazing, property, love... | custom porridge was provided to the baby brea... |
| 3 | 6.0 | 0.3017 | room, view, gateway, give, floor, high, small,... | allocated the numbernd room number which was ... |
| 4 | 1.0 | 0.1798 | staff, make, service, visit, hospitality, gues... | amazing experience if you are visiting mumbai... |
| 5 | 2.0 | 0.2500 | number, check, time, book, room, upgrade, thin... | taj should relook and compare yourself with y... |
| 6 | 0.0 | 0.1653 | experience, taj, feel, amazing, property, love... | every time i stay here i feel welcomes and lo... |
| 7 | 0.0 | 0.2921 | experience, taj, feel, amazing, property, love... | custom porridge was provided to the baby brea... |
| 8 | 6.0 | 0.2949 | room, view, gateway, give, floor, high, small,... | allocated the numbernd room number which was ... |
| 9 | 1.0 | 0.1827 | staff, make, service, visit, hospitality, gues... | amazing experience if you are visiting mumbai... |

### The most representative document for each topic:

Sometimes just the topic keywords may not be enough to make sense of what a topic is about. So, to help with understanding the topic, you can find the documents a given topic has contributed to the most and infer the topic by reading that document.

| Topic_Num | Topic_Perc_Contrib | Keywords | Text |
|---|---|---|---|
| 0.0 | 0.2921 | experience, taj, feel, amazing, property, love... | custom porridge was provided to the baby brea... |
| 1.0 | 0.2775 | staff, make, service, visit, hospitality, gues... | in my travels around the world i have never e... |
| 2.0 | 0.4436 | number, check, time, book, room, upgrade, thin... | the hospitality was missing and most of the s... |
| 3.0 | 0.3014 | hotel, pool, area, beautiful, nice, location, ... | had a wonderful three night stay at this hote... |
| 4.0 | 0.2847 | good, great, food, breakfast, restaurant, exce... | the overall stay was great few things to call... |

### Topic distribution across documents:

Finally, we want to understand the volume and distribution of topics in order to judge how widely it was discussed. The below table exposes that information.

| | Dominant_Topic | Topic_Keywords | Num_Documents | Perc_Documents |
|---|---|---|---|---|
| **0.0** | 2.0 | number, check, time, book, room, upgrade, thin... | 169.0 | 0.1693 |
| **1.0** | 0.0 | experience, taj, feel, amazing, property, love... | 139.0 | 0.1393 |
| **2.0** | 0.0 | experience, taj, feel, amazing, property, love... | 102.0 | 0.1022 |
| **3.0** | 6.0 | room, view, gateway, give, floor, high, small,... | 135.0 | 0.1353 |
| **4.0** | 1.0 | staff, make, service, visit, hospitality, gues... | 132.0 | 0.1323 |