Name: Marvin Nguyen

Table of Content

1. Executive Summary
2. Data Exploration
3. Sentiment Analysis
4. Topic Modeling

1. Executive Summary

The purpose of this investigation was to assist Airbnb in better understanding the opinions of its Albany, New York, listings. Identifying the most active reviewers, investigating review trends over time, using sentiment analysis to highlight important negative features of listings, and using topic modelling to Nnd the most often cited themes were our objectives. We give business owners useful insights by utilising artiNcial intelligence techniques, such as topic modelling with LDA and sentiment analysis with VADER on extracted word tokens. For instance, location and amenities were emphasised as key selling aspects, while preserving cleanliness and enhancing communication were noted as crucial elements. Strategic decisions to improve visitor pleasure and maximise corporate performance are directly supported by these insights.

2. Data Exploration

```
!pip install Cython
!pip install gensim
```

```
Requirement already satisfied: Cython in /usr/local/lib/python3.11/dist-pac
Requirement already satisfied: gensim in /usr/local/lib/python3.11/dist-pac
  Requirement already satisfied: numpy<2.0,>=1.18.5 in /usr/local/lib/python3
  Requirement already satisfied: scipy<1.14.0,>=1.7.0 in /usr/local/lib/pytho
  Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.
  Requirement already satisfied: wrapt in /usr/local/lib/python3.11/dist-pack
```

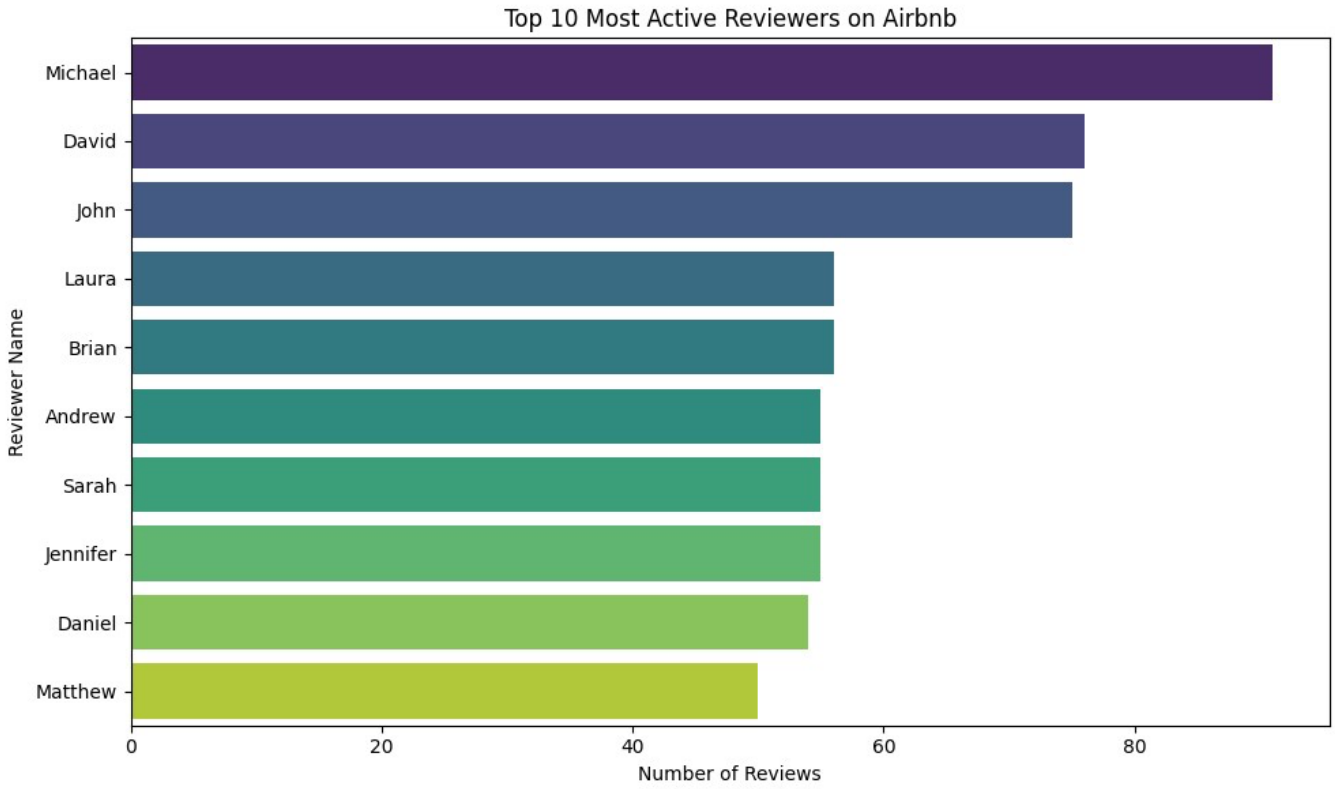
```
import pandas as pd

df = pd.read_excel('AirBnb_reviews.csv') df.head()
```

	listing_id	id	date	reviewer_id	reviewer_name	comments
0	2992450	35913434	2015-06-23	15772025	Jennifer	We were pleased to see how 2nd Street and the ...
1	3820211	19110185	2014-09-06	19139729	Harriett Jordan	Terra was a great host, super informative, and...

Next steps: Generate code with df View recommended plots New interactive sheet

```
import matplotlib.pyplot as plt
import seaborn as sns

# Identify the top 10 most active reviewers
top_reviewers = df['reviewer_name'].value_counts().nlargest(10)
print("Top 10 Most Active Reviewers:") print(top_reviewers)

# Bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x=top_reviewers.values, y=top_reviewers.index, palette='viridis')
plt.title('Top 10 Most Active Reviewers on Airbnb')
plt.xlabel('Number of Reviews')
plt.ylabel('Reviewer Name')
plt.tight_layout() plt.show()
```

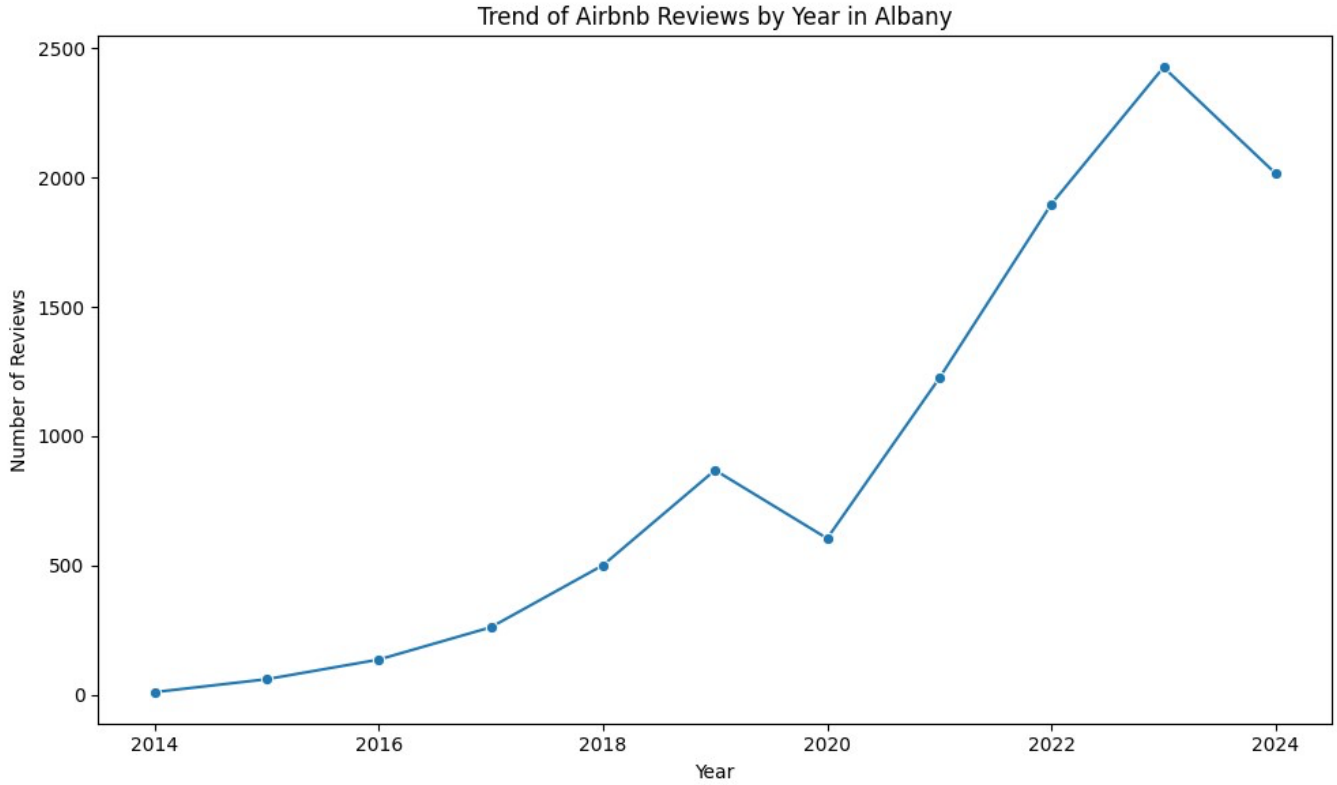
```
Top 10 Most Active Reviewers: reviewer_name
Michael     91
David       76
John        75
Laura       56
Brian       56
Andrew      55
Sarah       55
Jennifer    55
Daniel      54
Matthew     50
Name: count, dtype: int64 <ipython-input-3-c0c1eb931bba>:11: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed

sns.barplot(x=top_reviewers.values, y=top_reviewers.index, palette='virid
```



With 91 reviews, Michael is the most active reviewer according to the dataset. Brian (55), Laura (56), John (65), and David (70 reviews) are other very active reviewers. This suggests that a sizable portion of the dataset's total evaluations are contributed by a limited number of reviewers.

```
df['date'] = pd.to_datetime(df['date'], errors='coerce') df['year']
= df['date'].dt.year

reviews_by_year = df.groupby('year').size().reset_index(name='review_count')
print("Review Counts by Year:") print(reviews_by_year)

# Line plot
plt.figure(figsize=(10, 6))
sns.lineplot(data=reviews_by_year, x='year', y='review_count', marker='o')
plt.title('Trend of Airbnb Reviews by Year in Albany')
plt.xlabel('Year')
plt.ylabel('Number of Reviews')
plt.tight_layout() plt.show()
```

```
Review Counts by Year:
    year    review_count  0
    2014             9
1   2015              59
2   2016             135
3   2017             260
4   2018             500
5   2019             868
6   2020             604
7   2021            1224
8   2022            1898
9   2023            2427
10  2024            2016
```



From just 9 reviews in 2014 to over 2,400 in 2023, the review counts clearly demonstrate an increased trend, suggesting that Albany has grown in popularity as an Airbnb destination. COVID-19 limits are probably to blame for the 2020 decline, which was followed by a robust recovery. Airbnb's growing popularity, e^cient marketing, Albany's status as a capital hosting government events, and a post-pandemic spike in demand for travel are some of the factors propelling this trend.

3. Sentiment Analysis

```
from nltk.stem import PorterStemmer
import re  import
nltk
```

```
nltk.download('stopwords') from
nltk.corpus import stopwords

porter = PorterStemmer()
stop_words = stopwords.words('english')

documents = df['comments'] cleaned_docs

= []

for review in documents:
    try:
        cleaned = re.sub('[^A-Za-z]', ' ', review)
        cleaned = cleaned.lower()

        tokens = cleaned.split()

        tokens = [porter.stem(word) for word in tokens]

        tokens = [word for word in tokens if len(word) > 3]

        cleaned_review = ' '.join(tokens)
    except Exception as e:
        print(f"Error processing review: {e}")
        cleaned_docs.append('')
        continue

    cleaned_docs.append(cleaned_review)     print('-
[Review Text]: ', cleaned_review) final_docs = []

for doc in cleaned_docs:
    filtered_tokens = [word for word in doc.split() if word not in stop_words]
    final_doc = ' '.join(filtered_tokens)
    final_docs.append(final_doc)
    print('-[Cleaned Text]: ', final_doc)

df['cleaned_comments'] = final_docs
-[Cleaned Text]:  remind home beauti architectur around perfect stay great
-[Cleaned Text]:  beauti beauti build best part albani stay airbnb price wo
-[Cleaned Text]:  stay david place thanksgiv retreat place spaciou clean li
-[Cleaned Text]:  beauti suit everyth exactli list enjoy stay would stay da
-[Cleaned Text]:  amaz place stay full beauti david pleasur speak respond q
-[Cleaned Text]:  amaz stay super warm winter great neighborhood excel host
-[Cleaned Text]:  believ luck book space right gild fresh delight everi tur
-[Cleaned Text]:  neat place cool
-[Cleaned Text]:  thank great stay
-[Cleaned Text]:  absolut beauti inspir place stay magnific parlor excel pa
-[Cleaned Text]:  amaz rent restor victorian space histor district close ve
-[Cleaned Text]:  love stay david place home base dure advocaci visit state
-[Cleaned Text]:  nice central locat safe nice place
```

```
-[Cleaned Text]:  love david place neighborhood great feel veri safe amazin
-[Cleaned Text]:  histor home describ proxim event attend access courtyard
-[Cleaned Text]:  david except commun respond inquiri immedi veri friendli
-[Cleaned Text]:  stay night uniqu airbnb enjoy space locat shower water pr
-[Cleaned Text]:  beauti place perfect locat love histor charact charm davi
-[Cleaned Text]:  beauti hous charact well locat appoint
-[Cleaned Text]:  cool place histor modern brand central bath veri updat co
-[Cleaned Text]:  beauti stay locat perfect suit expans would gladli stay
-[Cleaned Text]:  space great bill comfort first floor histor brownston his
-[Cleaned Text]:  realli great place albani
-[Cleaned Text]:  perfect spot famili visit slater beauti tall eleg intent
-[Cleaned Text]:  great time albani would definit stay back town highli rec
-[Cleaned Text]:  great place close everyth origin thought decor veri uniqu
-[Cleaned Text]:  veri nice larg place
-[Cleaned Text]:  place beauti pictur justic david easi deal veri respons g
-[Cleaned Text]:  love space everyth want commun easi perfect spot
-[Cleaned Text]:  david histor place spaciou comfort especi love galleri at
-[Cleaned Text]:  apart clean quiet walkabl washington park empir state pla
-[Cleaned Text]:  great place privaci weekend getaway almost everyth need a
-[Cleaned Text]:  pleasur stay abba place glad stay return hometown
-[Cleaned Text]:  abba made check check process extrem easi clean would lik
-[Cleaned Text]:  great place stay near univers abba wonder instruct last m
-[Cleaned Text]:  good stay littl abba sent someon
-[Cleaned Text]:  realli enjoy space love littl room front window everyth w
-[Cleaned Text]:  wonder place stay home cozi spotless exceed expect store
-[Cleaned Text]:  ermenita place littl comfort nest quiet citi albani short
-[Cleaned Text]:  veri love home definit return
-[Cleaned Text]:  ermenita place perfect unwind relax trip close airport th
-[Cleaned Text]:  love stay back thank ermenita
-[Cleaned Text]:  visit brief night ermenita place wonder comfort hous char
-[Cleaned Text]:  home beauti spaciou describ although issu host respons ca
-[Cleaned Text]:  highest compliment give back quiet comfort even well appo -
 [Cleaned Text]:  charm thought appoint home perfect famili need ermenita w
-[Cleaned Text]:  littl green hous describ comfort welcom privat park back
-[Cleaned Text]:  gorgeou home albani briefli earli meet capitol build easi
-[Cleaned Text]:  home beauti histor charm room great place feel cozi relax
-[Cleaned Text]:  beauti hous describ thought detail host respons kind love
-[Cleaned Text]:  great hous enjoy time everyth need thank
-[Cleaned Text]:  love littl green hous felt veri homey comfort thank thoug
-[Cleaned Text]:  veri cosi welcom home well furnish stock host extrem comm
-[Cleaned Text]:  wonder place excel locat close everyth main drive scotlan
-[Cleaned Text]:  great stay littl green hous onli town night bruce springs
-[Cleaned Text]:  great host respons proactiv help great venu well equip ki
-[Cleaned Text]:  wonder experi beauti decor home super clean comfort enjoy
-[Cleaned Text]:  beauti place expect host veri respons would definit visit
-[Cleaned Text]:  ermenita hous beauti space comfort soon walk furnish eleg
-[Cleaned Text]:  wonder littl hous beauti neighborhood albani came town la
```

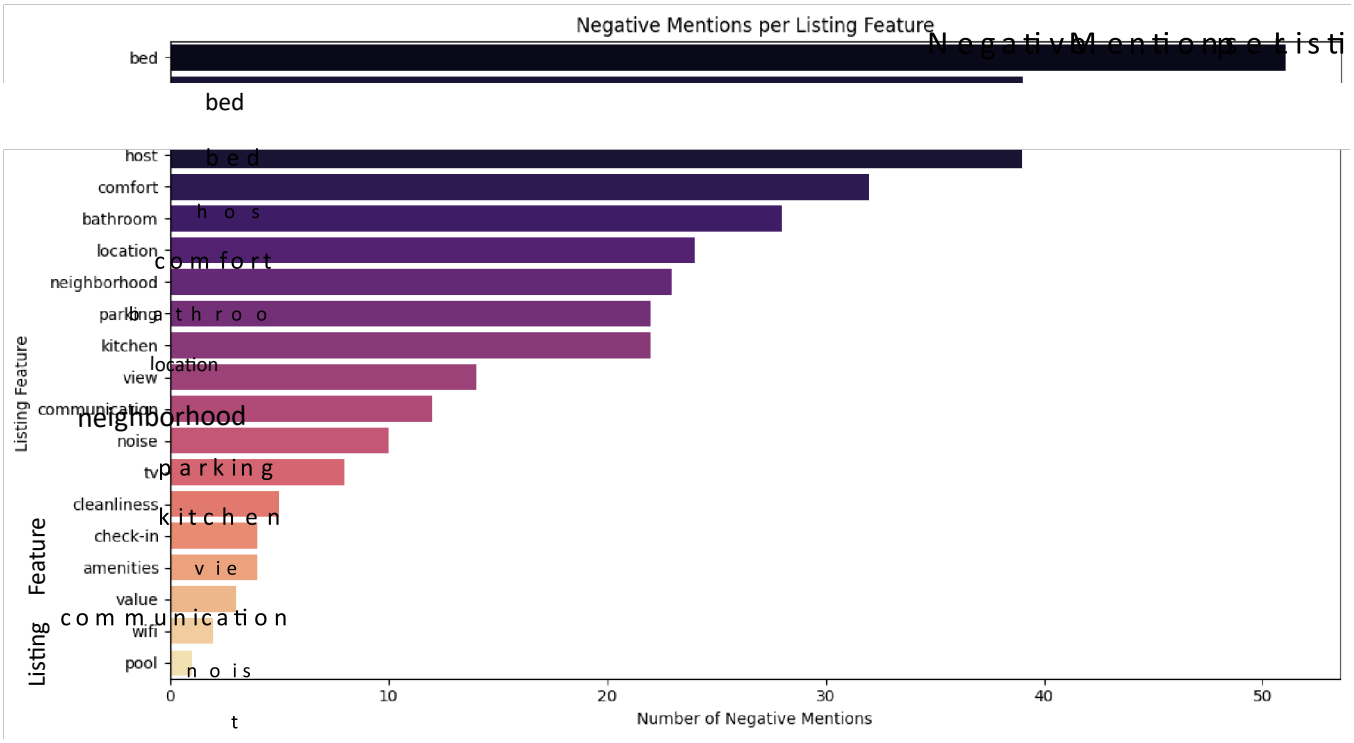
```
import nltk

from nltk.sentiment import SentimentIntensityAnalyzer

import matplotlib.pyplot as plt import seaborn as sns
```

```
nltk.download('vader_lexicon') df['comments'] =

df['comments'].astype(str)

sia = SentimentIntensityAnalyzer()
df['sentiment_score'] = df['comments'].apply(lambda text: sia.polarity_scores(t

df['sentiment_label'] = df['sentiment_score'].apply(lambda score: 'negative' if

else ('positive' if score print(df[['comments', 'sentiment_score',

'sentiment_label']].head())
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
                                                comments  sentiment_score  \ 0
     We were pleased to see how 2nd Street and the ...           0.9883
     1  Terra was a great host, super informative, and...           0.9768
     2  The apartment was clean, nice & convenient. I ...           0.9150
     3  Very comfortable apartment and super central t...           0.9682   4
        Our favourite place to stay in Albany! The pla...           0.9813

       sentiment_label  0
     positive
     1        positive
     2        positive
     3        positive
     4        positive
```

```
features = [
    "location", "cleanliness", "comfort", "host", "neighborhood",
    "noise", "value", "amenities", "check-in", "communication",
    "wifi", "parking", "view", "bed", "bathroom", "kitchen", "pool", "tv"
]

feature_negative_counts = {}
for feature in features:
    mask = df['comments'].str.contains(feature, case=False, na=False)
    negative_count = df[mask & (df['sentiment_label'] == 'negative')].shape[0]
    feature_negative_counts[feature] = negative_count

import pandas as pd
  feature_negative_df = pd.DataFrame.from_dict(feature_negative_counts, orient='i
  feature_negative_df = feature_negative_df.sort_values(by='Negative Mentions', a
print("Negative Mentions by Feature:") print(feature_negative_df)

plt.figure(figsize=(12, 6))
sns.barplot(x=feature_negative_df['Negative Mentions'], y=feature_negative_df.i
plt.title("Negative Mentions per Listing Feature")
plt.xlabel("Number of Negative Mentions")
plt.ylabel("Listing Feature")
plt.tight_layout() plt.show()
```

```
Negative Mentions by Feature:
                Negative Mentions bed                            51 host
39 comfort                        32 bathroom                       28
location                        24 neighborhood                   23 parking
22 kitchen                        22 view                           14
communication                   12 noise                          10 tv
8 cleanliness                     5 check-in                        4
amenities                        4 value                           3 wifi
2 pool                            1 <ipython-input-7-f8e443876119>:20:
FutureWarning: Passing `palette` without assigning `hue` is deprecated and
will be removed   sns.barplot(x=feature_negative_df['Negative Mentions'],
y=feature_negativ
```



I utilized VADER to assess sentiment scores in my analysis. The overarching positive sentiment score indicates that visitors generally feel satisNed with their Airbnb experiences. However, despite this generally optimistic outlook, there is a notable presence of negative feedback pertaining to speciNc aspects, such as comfort (32), host communication (39), bed quality (51), and toilet conditions (28). This suggests that although the overall

impressions are positive, enhancing these critical areas could further elevate visitor satisfaction.

4. Topic Modeling

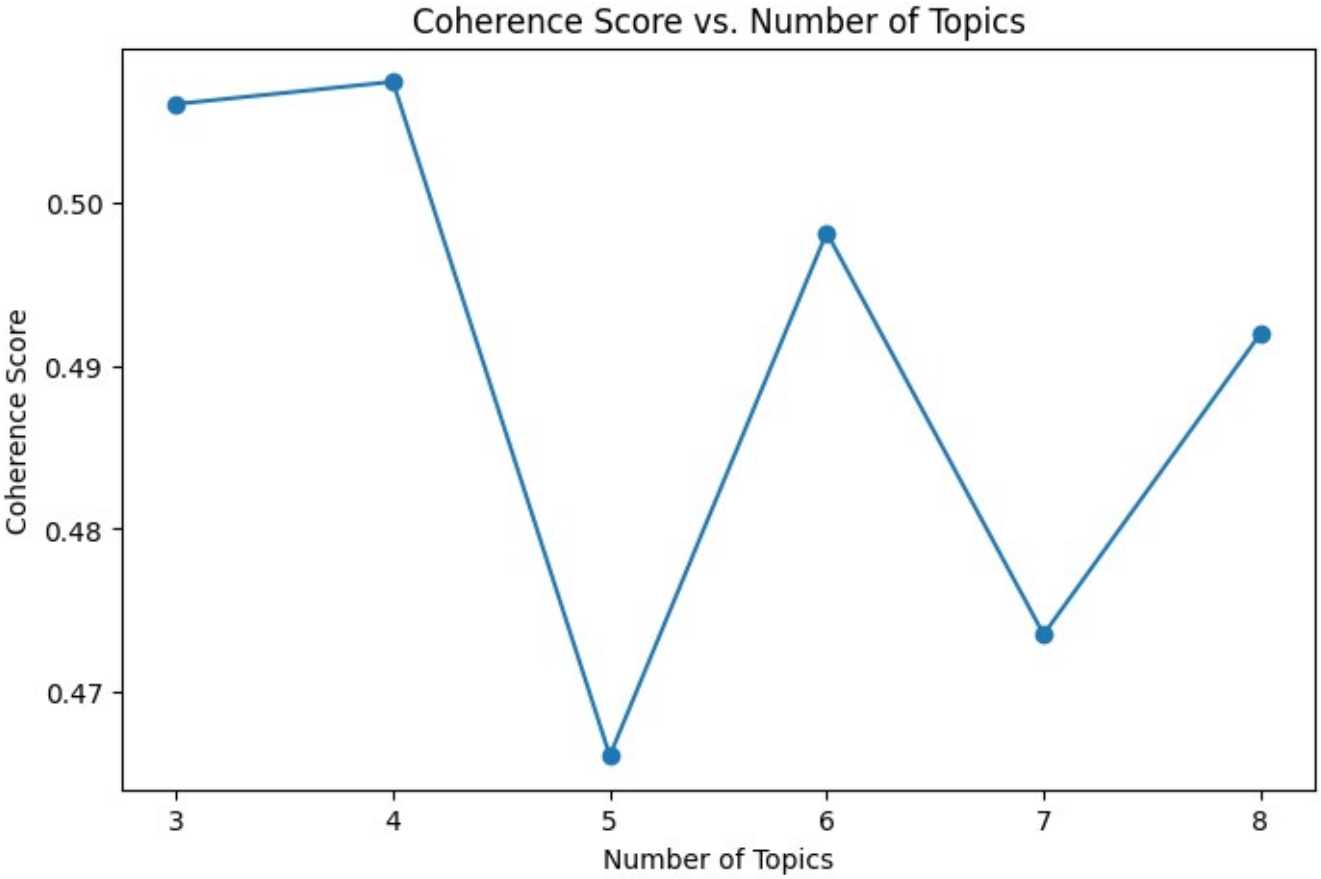
```
import re
import nltk
from nltk import pos_tag, word_tokenize from
nltk.corpus import stopwords

nltk.download('punkt_tab')
nltk.download('averaged_perceptron_tagger_eng')

nltk.download('stopwords') stop_words =

set(stopwords.words('english'))

def extract_nouns(text):
    """
    Cleans text, tokenizes, performs POS tagging, and extracts only nouns.
    Filters out stop words and words shorter than 4 characters.
    """

    text = re.sub('[^A-Za-z]', ' ', text.lower())
    tokens = word_tokenize(text)

    nouns = [word for word, pos in pos_tag(tokens) if pos.startswith('NN')]

    nouns = [word for word in nouns if word not in stop_words and len(word) > 3
    return nouns


df['noun_tokens'] = df['comments'].astype(str).apply(extract_nouns) df

= df[df['noun_tokens'].map(len) > 0]
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
     [nltk_data] Downloading package averaged_perceptron_tagger_eng to
     [nltk_data]     /root/nltk_data... [nltk_data]   Package
     averaged_perceptron_tagger_eng is already up-to-
     [nltk_data]       date!
     [nltk_data] Downloading package stopwords to /root/nltk_data...
     [nltk_data]   Package stopwords is already up-to-date!
```

```
from gensim import corpora dictionary =

corpora.Dictionary(df['noun_tokens'])
```

```
dictionary.filter_extremes(no_below=5, no_above=0.5) corpus =

[dictionary.doc2bow(text) for text in df['noun_tokens']]


from gensim.models import LdaModel
from gensim.models import LdaModel
from gensim.models.coherencemodel import CoherenceModel import
matplotlib.pyplot as plt

topic_range = range(3, 9)
coherence_values = []
model_list = []

for num_topics in topic_range:
    lda_model = LdaModel(corpus=corpus, id2word=dictionary, num_topics=num_topic
                         passes=10, random_state=42)
    model_list.append(lda_model)
    coherence_model = CoherenceModel(model=lda_model, texts=df['noun_tokens'],
                                     dictionary=dictionary, coherence='c_v')
    coherence = coherence_model.get_coherence()
    coherence_values.append(coherence)
    print(f"Num Topics = {num_topics} --> Coherence Score = {coherence:.4f}")

plt.figure(figsize=(8, 5))
plt.plot(topic_range, coherence_values, marker='o')
plt.xlabel("Number of Topics") plt.ylabel("Coherence
Score")
plt.title("Coherence Score vs. Number of Topics")
plt.show()
```

```
Num Topics = 3 --> Coherence Score = 0.5060
Num Topics = 4 --> Coherence Score = 0.5074
Num Topics = 5 --> Coherence Score = 0.4661
Num Topics = 6 --> Coherence Score = 0.4981
Num Topics = 7 --> Coherence Score = 0.4735
Num Topics = 8 --> Coherence Score = 0.4920
```




```
for idx, topic in topics:


optimal_topics = topic_range[coherence_values.index(max(coherence_values))]

print(f"\nOptimal number of topics determined: {optimal_topics}") optimal_model

= model_list[coherence_values.index(max(coherence_values))]

print("\nDiscovered Topics:")
topics = optimal_model.print_topics(num_words=10) for
idx, topic in topics:
    print(f"Topic {idx+1}: {topic}")
```

```
Optimal number of topics determined: 4

     Discovered Topics:
     Topic 1: 0.042*"room" + 0.028*"kitchen" + 0.019*"bathroom" + 0.016*"house"
     Topic 2: 0.104*"stay" + 0.095*"apartment" + 0.060*"location" + 0.031*"stree
     Topic 3: 0.066*"home" + 0.050*"host" + 0.044*"stay" + 0.036*"house" + 0.029
     Topic 4: 0.266*"place" + 0.043*"host" + 0.035*"location" + 0.033*"everythin
```

In the analysis, the four-topic model came out on top with the highest coherence score—about 0.5074. This means it struck the best balance between having distinct themes and being easy to understand. While the three-topic model also did a good job, the four-topic approach gave us a little more detail without breaking the content into too many pieces.

The four topics we uncovered tell an interesting story. One topic zeroes in on amenities and comfort, focusing on the little in-room features that make a stay cozy. Another topic looks at location and practical details like how close things are and the availability of parking. A third topic highlights the role of the host and local experience, showcasing how hospitality and local recommendations can enrich a stay. Lastly, the fourth topic centers on host communication and the check-in process, which are crucial for a smooth experience. Overall, these insights offer clear, actionable ideas for boosting guest satisfaction and improving business performance.

```
!pip install wordcloud from
wordcloud import WordCloud
import math

rows_wc = math.ceil(optimal_topics / 4)
fig, ax = plt.subplots(rows_wc, 4, figsize=(15, 2.5 * rows_wc))

if rows_wc == 1:
    ax = ax.reshape(1, -1)

[axi.set_axis_off() for axi in ax.ravel()]

topics_wc = optimal_model.show_topics(num_topics=optimal_topics, num_words=20, for
topic_idx, topic in enumerate(topics_wc):

    word_freq = dict(topic[1][:10])
    wordcloud = WordCloud(background_color="white").generate_from_frequencies(w
    row_idx = topic_idx // 5
    col_idx = topic_idx % 5
    ax[row_idx, col_idx].imshow(wordcloud)
    ax[row_idx, col_idx].set_title(f"Topic {topic_idx+1}")
plt.tight_layout() plt.show()
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.11/dist-
 Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.11/di
 Requirement already satisfied: pillow in /usr/local/lib/python3.11/dist-pac
 Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist
 Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.1
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/di
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.
 Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.
 Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11
 Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.1
 Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/pytho
```

```
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-p
```

Topic1

bathroomhouseapartment lran

spotstreet

albanySTaV

hom

kitchenstreeticationarke

```
pinterpretation = """
Topic 1 - Amenities & Comfort was characterized by words such as "room," "kitch
For this reason, business owners must improve the quality and the description o

Topic 2 - Location & Stay Factors has high counts of "stay," "apartment," "loca
where the property is and how accessible it is. Hosts could emphasize nearby la

Topic 3 - Host & Local Experience has high-frequency terms such as "home," "hos
host personality are very significant to guests. By way of response, hosts may

Topic 4 - Host Communication & Check-in is governed by words such as "host," "c
smooth check-in are the requirements. Therefore, business owners should ensure
""" print(pinterpretation)
```

```
Topic 1 - Amenities & Comfort was characterized by words such as "room," "k
     For this reason, business owners must improve the quality and the descripti

     Topic 2 - Location & Stay Factors has high counts of "stay," "apartment," "
     where the property is and how accessible it is. Hosts could emphasize nearb

     Topic 3 - Host & Local Experience has high-frequency terms such as "home,"
     host personality are very significant to guests. By way of response, hosts

     Topic 4 - Host Communication & Check-in is governed by words such as "host,
     smooth check-in are the requirements. Therefore, business owners should ens
```

I found a number of important themes based on our topic modelling research, which involved extracting just nouns from the English reviews using text processing. The subjects that are most often brought up are:

- Location & Neighbourhood: Visitors often talk about things like the property's location, how close it is to attractions, and the general atmosphere of the neighbourhood.
- Cleaning & upkeep: A lot of evaluations stress how crucial cleaning and regular upkeep are.

- Amenities & Comfort: Remarks frequently highlight the standard of amenities such as kitchenware, wiN, and general comfort when visiting.
- Host Interaction & Communication: Evaluations stress the importance of responsive hosting, easy check-in procedures, and e^cient communication.

Although pricing did not surface as a separate topic in the model, the prevalence of negative mentions and recurring review patterns indicate that it remains a notable concern Price is a sensitive topic for guests, who often discuss whether they felt they got a good deal.

Suggestions:

In order to enhance their business e^ciency, Airbnb hosts ought to:

- Improve Local Listings: To capitalise on the interest in location, include comprehensive information about the neighbourhood and neighbouring attractions.
- Invest in Cleaning Protocols: To address concerns about cleanliness, implement strict cleaning and maintenance routines.
- Enhance Amenities: Consistently update the property's features and make sure that its top-notch amenities are prominently displayed.
- Enhance Host Training: Put an emphasis on e^cient check-in processes and good communication.
- Optimise Pricing Strategies: Take into account dynamic pricing models and make sure that the value being supplied is communicated effectively in the pricing speciNcs.