```
import pandas as pd
import numpy as np
import plotly.express as px
```

Dataset Overview:

- The guest_data_with_reviews.xlsx dataset contains customer feedback from a hospitality business.
- Key columns include: How likely are you to recommend us to a friend or colleague? for Net Promoter Score (NPS) calculations. Review for textual feedback, useful for sentiment analysis and topic modeling.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df=pd.read_excel("/content/guest_data_with_reviews.xlsx")
```

```
df.head()
```

	ID	Start time	Completion time	Email	Name	Full Name	Gender	Date of Birth	Checkout Date	Purpose of the visit	...	How likely are you to recommend us to a friend or colleague?	Staff attitude	Check-in Process	Room service	F cleanlir
0	1	NaN	NaN	NaN	NaN	Guest 00001	Male	1993-10-02	2022-04-07	Business	...	9	Good	Good	Very good	I
1	3	NaN	NaN	NaN	NaN	Guest 00003	Male	1981-10-03	2020-01-16	Vacation	...	4	Poor	Good	Very good	Ave

```
#we can see some columns might have missing value. we are looking for calculating the NPS score.
```

```
# How many rows  in the dataset
total_len=len(df)
total_len
```

```
1108
```

```
#Missing value
missing_values = df.isnull().sum()
print(missing_values)
```

```
ID                                                                  0
Start time                                                       1108
Completion time                                                  1108
Email                                                            1108
Name                                                             1108
Full Name                                                           0
Gender                                                              0
Date of Birth                                                       0
Checkout Date                                                       0
Purpose of the visit                                                0
How did you discover us?                                            0
Rate your overall experience in our hotel                           0
How likely are you to recommend us to a friend or colleague?        0
Staff attitude                                                      0
Check-in Process                                                    0
Room service                                                        0
Room cleanliness                                                    0
Food quality                                                        0
Variety of food                                                     0
Broadband & TV                                                      0
Gym                                                                 1
Review                                                              0
dtype: int64
```

```
#we can see Start time,Completion time, Name, Email, Full Name >> we dont have any value. we assume that is fine bcz we
```

Missing Value Treatement

```
#We have 1 miising value in the Gym indicator#
# here will not use dropna()>> bcz kit will completely detele missing rows in any of the columns

#1st drop the column and then rows with missing value.
df_cleaned= df.dropna(axis=1)
df_cleaned
```

	ID	Full Name	Gender	Date of Birth	Checkout Date	Purpose of the visit	How did you discover us?	Rate your overall experience in our hotel	How likely are you to recommend us to a friend or colleague?	Staff attitude	Check-in Process	Room service	Room cleanliness	qu
0	1	Guest 00001	Male	1993-10-02	2022-04-07	Business	Organization	3	9	Good	Good	Very good	Poor	
1	3	Guest 00003	Male	1981-10-03	2020-01-16	Vacation	News paper	4	4	Poor	Good	Very good	Average	
2	4	Guest 00004	Male	2004-03-31	2022-05-14	Vacation	Search engine	5	6	Good	Very good	Good	Poor	A
3	5	Guest 00005	Male	1961-08-08	2022-06-24	Business	hotel booking sites	5	5	Excellent	Excellent	Very good	Average	
4	8	Guest 00008	Male	1981-11-27	2020-02-01	Business	Organization	1	7	Good	Excellent	Good	Average	Ex

Next steps: View recommended plots New interactive sheet

Key Concepts: Net Promoter Score (NPS): Measures customer loyalty. Scores:

- 9–10: Promoters
- 7–8: Passives
- 0–6: Detractors

Formula: NPS = percentage of promoters - percentage of detractors

- Sentiment Analysis: Identifies the emotional tone (positive, negative, or neutral) in reviews.
- Topic Modeling: Uses embeddings to identify recurring themes in text data.

```
#How likely are you recommend us?
likelyhood= px.histogram(df_cleaned, x="How likely are you to recommend us to a friend or colleague?",nbins=10,title="[

likelyhood.show()
```


```
#calculate NPS score:

#classify score Promoter(9-10), passive (7-8), Detractor(0-6)

#We have to create a function to define the NPs score:

def classify_NPS(Score):
  if Score>=9:
    return 'Promoters'
  elif Score>=7:
    return 'Passive'
  else:
    return 'Detractor'

#Apply the classification on the recommendation score

df_cleaned['NPS Score']=df_cleaned["How likely are you to recommend us to a friend or colleague?"].apply(classify_NPS)

df_cleaned
```

```
<ipython-input-12-c84b93280c00>:3: 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#returning-a-view-versus
```

	ID	Full Name	Gender	Date of Birth	Checkout Date	Purpose of the visit	How did you discover us?	Rate your overall experience in our hotel	How likely are you to recommend us to a friend or colleague?	Staff attitude	Check-in Process	Room service	Room cleanliness	qu
0	1	Guest 00001	Male	1993-10-02	2022-04-07	Business	Organization	3	9	Good	Good	Very good	Poor	
1	3	Guest 00003	Male	1981-10-03	2020-01-16	Vacation	News paper	4	4	Poor	Good	Very good	Average	
2	4	Guest 00004	Male	2004-03-31	2022-05-14	Vacation	Search engine	5	6	Good	Very good	Good	Poor	A
3	5	Guest 00005	Male	1961-08-08	2022-06-24	Business	hotel booking sites	5	5	Excellent	Excellent	Very good	Average	
4	8	Guest 00008	Male	1981-11-27	2020-02-01	Business	Organization	1	7	Good	Excellent	Good	Average	Ex

Next steps: View recommended plots New interactive sheet

```
#Ratio of Promoters, Dectractor and Passive
NPS_proportions= df_cleaned["NPS Score"].value_counts(normalize=True)*100
NPS_proportions
```

NPS Score	proportion
Passive	47.202166
Detractor	36.913357
Promoters	15.884477

```
#As Larger num of Promoters than Dectractors. So, we can say more negative number of feedback.Business is not going vwe
```

```
#Interprate and claculate the overall NPS
Promoters=df_cleaned[df_cleaned["NPS Score"]=="Promoters"].shape[0]
Detractors=df_cleaned[df_cleaned["NPS Score"]=="Detractor"].shape[0]
Passive=df_cleaned[df_cleaned["NPS Score"]=="Passive"].shape[0]
Total_response=df_cleaned.shape[0]
```

```
Promoters, Detractors, Passive, Total_response
```

```
(176, 409, 523, 1108)
```

```
nps_score=((Promoters-Detractors)/Total_response)*100
nps_score
```

```
-21.028880866425993
```

```
if nps_score>0:
  nps_interpretation="Positive"
```

```
elif nps_score<0:
  nps_interpretation="Negative"
else:
  nps_interpretation="Neutral"
```

```
nps_score,nps_interpretation
```

```
(-21.028880866425993, 'Negative')
```

```
# What does NPS value indicate about customer loyalty?
if nps_score>50:
  loyalty_interpretation="Excellent customer loyalty"
elif 0 <nps_score<=50:
  loyalty_interpretation="Good customer loyalty"
else:
  loyalty_interpretation="very poor customer loyalty"
```

```
nps_score,nps_interpretation, loyalty_interpretation
```

```
(-21.028880866425993, 'Negative', 'very poor customer loyalty')
```

```
#Sentiment Analysis

from transformers import pipeline


sentiment_analzer = pipeline("sentiment-analysis")

#Extracts the first prediction ([0]) and gets its sentiment label (['label']).
df_cleaned['sentiment']=df_cleaned['Review'].apply(lambda review:sentiment_analzer(review)[0]['label'])
```

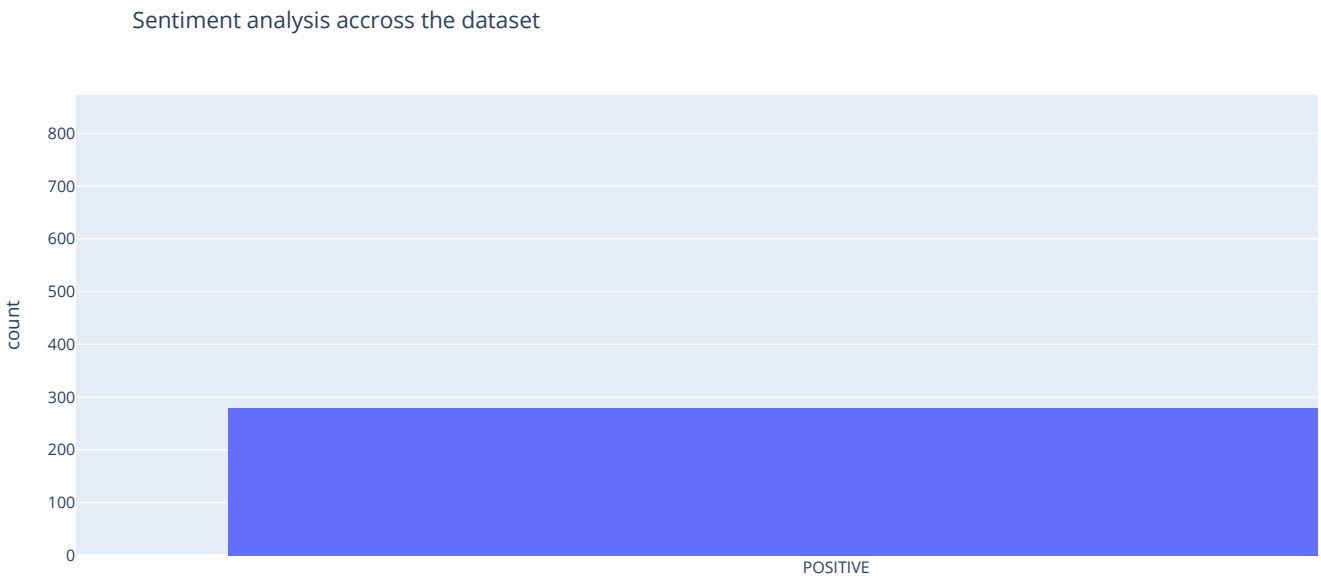
```
No model was supplied, defaulted to distilbert/distilbert-base-uncased-finetuned-sst-2-english and revision 714eb0f (https://huggin
Using a pipeline without specifying a model name and revision in production is not recommended.
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:


The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as 
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.

config.json: 100%                                   629/629 [00:00<00:00, 25.7kB/s]
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better p
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back
model.safetensors: 100%                             268M/268M [00:01<00:00, 199MB/s]

tokenizer_config.json: 100%                           48.0/48.0 [00:00<00:00, 3.70kB/s]

vocab.txt: 100%                                     232k/232k [00:00<00:00, 6.32MB/s]
Device set to use cpu
<ipython-input-23-cc57db3347c5>:4: 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#returning-a-view-versus
```

```
df_cleaned[['Review','sentiment']]
```

	Review	sentiment
0	it was an okay experience overall. the room wa...	POSITIVE
1	the stay was memorable thanks to the poor staf...	NEGATIVE
2	while the staff were good and did their best t...	NEGATIVE
3	the hotel experience was average at best. the ...	NEGATIVE
4	not my favorite stay, but the good staff helpe...	POSITIVE
...	...	...
1103	i was pleasantly surprised by how excellent th...	NEGATIVE
1104	the hotel experience was average at best. the ...	NEGATIVE
1105	this stay was a mix of good and bad. the staff...	NEGATIVE
1106	there's room for improvement here. while the s...	NEGATIVE
1107	not my favorite stay, but the excellent staff ...	POSITIVE

1108 rows x 2 columns

```
#Visualize sentiment analysis accross the distribution:

fig= px.histogram(df_cleaned,x="sentiment",
                 title= "Sentiment analysis accross the dataset")


fig.show()
```




```
sentiment_count=df_cleaned['sentiment'].value_counts()
sentiment_percentage=(df_cleaned['sentiment'].value_counts())/len(df_cleaned)*100

sentiment_percentage
```

sentiment	count
NEGATIVE	74.819495
POSITIVE	25.180505

```
#prepare a dataframe to show :

sentiment_df = pd.DataFrame({'Sentiment': sentiment_count.index,
                             'count': sentiment_count.values,
                             'percentage': sentiment_percentage.values})


sentiment_df
```

	Sentiment	count	percentage
0	NEGATIVE	829	74.819495
1	POSITIVE	279	25.180505

```
#what are the common keywords in the review:
#CountVectorizer >>help to count

from sklearn.feature_extraction.text import CountVectorizer
Review= df_cleaned['Review']

vectorizer= CountVectorizer(stop_words='english', max_features=20)

#Fit the data:

X= vectorizer.fit_transform(Review)

get_vocab= vectorizer.get_feature_names_out()

word_count=X.toarray().sum(axis=0)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-4-a4e8b31d06f0> in <cell line: 0>()
      3
      4 from sklearn.feature_extraction.text import CountVectorizer
----> 5 Review= df_cleaned['Review']
      6
      7 vectorizer= CountVectorizer(stop_words='english', max_features=20)

NameError: name 'df_cleaned' is not defined
```

```
#create a dataframe with word and count

keywords_df= pd.DataFrame({
    'keyword': get_vocab,
    'count': word_count
}).sort_values(by='count', ascending= False)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-1316e216ed43> in <cell line: 0>()
      2
      3 keywords_df= pd.DataFrame({
----> 4     'keyword': get_vocab,
      5     'count': word_count
      6 }).sort_values(by='count', ascending= False)

NameError: name 'get_vocab' is not defined
```

```
keywords_df
```

	keyword	count
15	room	1108
8	food	1003
9	good	997
16	staff	897
17	stay	457
7	experience	454
3	cleanliness	403
2	best	314
5	excellent	307
19	wasn	306
1	average	246
12	overall	245
4	didn	220
14	quality	217
18	team	211
6	expectations	203
10	like	194
13	poor	163
0	approach	149
11	make	149

Next steps: View recommended plots New interactive sheet

TF-IDF vectorization and Non-negative Matrix Factorization (NMF) help to extract key topics

- TfidfVectorizer: Converts text data into a matrix of TF-IDF (Term Frequency-Inverse Document Frequency) features
- NMF: A machine learning technique used for topic modeling (identifying hidden topics in text).
- n_components=5: The model will extract 5 topics.
- Trains the NMF model on the TF-IDF matrix to identify 5 topics based on word distributions.
- Returns a matrix where each row represents a topic, and each column represents the importance of a word in that topic.
- List item

```
from sklearn.decomposition import LatentDirichletAllocation
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import NMF


#using emedding extract key topics
# Initialize the TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_features=1000)

# Fit and transform the reviews
tfidf = tfidf_vectorizer.fit_transform(Review)

#Initialize the NMF Model

nmf=NMF(n_components=5, random_state=42)

#Fit the NMF
nmf.fit(tfidf)

#get the topic

topic=nmf.components_

#Get the feature names(Words)
feature_names=tfidf_vectorizer.get_feature_names_out()

#Dispaly the top words for each topic
num_topword=10
```

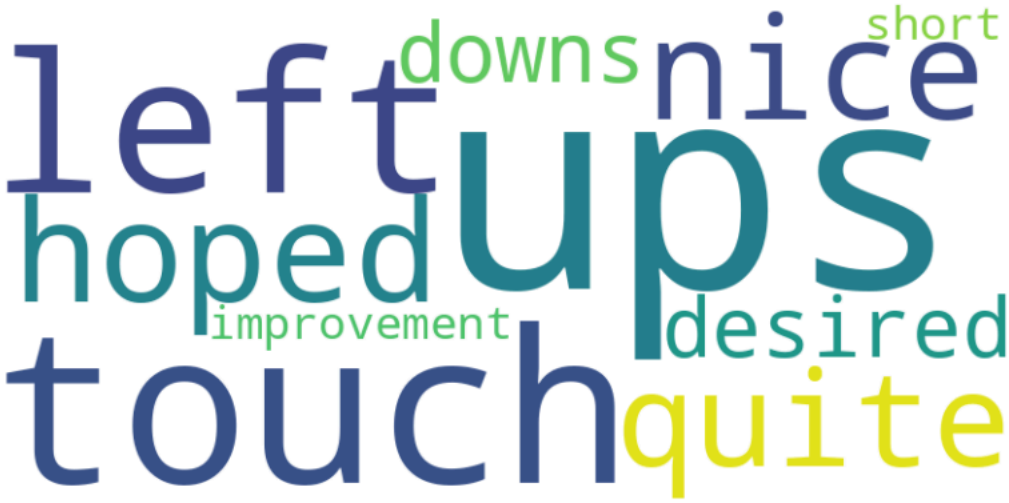
```
topics_df=pd.DataFrame()

for topic_idx, topic in enumerate(topic):
  top_word=[feature_names[i] for i in topic.argsort()[:-num_topword -1:-1]]
  topics_df[f'Topic{topic_idx+1}'] = top_word
topics_df
```

	Topic1	Topic2	Topic3	Topic4	Topic5
0	expectations	upgrades	overall	best	wasn
1	cleanliness	consider	work	experience	good
2	memorable	helped	tried	dining	impression
3	match	need	make	improved	liked
4	thanks	returning	okay	worst	pleasantly
5	unfortunately	significant	use	did	surprised
6	quality	favorite	approach	assist	leave
7	didn	stay	bit	hope	clean
8	stay	staff	experience	hotel	quite
9	staff	food	good	future	left

Next steps: View recommended plots New interactive sheet

```
# Visualize results as a word cloud

from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Function to plot word cloud for each topic
def plot_word_cloud(lda_model, feature_names, num_top_words):
    for topic_idx, topic in enumerate(lda_model.components_):
        word_freq = {feature_names[i]: topic[i] for i in topic.argsort()[:-num_top_words - 1:-1]}
        wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_frequencies(word_freq)

        plt.figure(figsize=(10, 5))
        plt.imshow(wordcloud, interpolation='bilinear')
        plt.axis('off')
        plt.title(f'Topic {topic_idx+1}')
        plt.show()

# Plot word clouds for each topic
plot_word_cloud(lda, tfidf_feature_names, no_top_words)
```

Topic 1



Topic 2