Name: Ashish Gupta

Roll No.: 16

Class: D16AD

Scrapped data using Instant Data Scaper Google Extention and saved scraped data as as .csv file

```
import pandas as pd

file_path = 'google.csv'

df = pd.read_csv(file_path)
```

```
df.head(2)
```

	d4r55	fzvQIb	xRkPPb	qmhsmd	wiI7pd
0	ASHITHA PS	5/5	3 weeks ago on	Google	I stayed at LA Hotels metro when on a recent h...
1	Umair Rashidi	5/5	a day ago on	Google	LA Hotel Metro exceeded all expectations! Impe...

```
new_column_names = ['Name', 'Rating', 'Date', 'Platform', 'Review']

df.columns = new_column_names

# Remove numbering from the 'Rating' and 'Date' columns
df['Rating'] = df['Rating'].str.replace(r'\d/5', '', regex=True)
df['Date'] = df['Date'].str.replace(r'\d+ (day|week|weeks|month|months|year|years) ago on', '', regex=True)
```

```
df.head(2)
```

	Name	Rating	Date	Platform	Review
0	ASHITHA PS			Google	I stayed at LA Hotels metro when on a recent h...
1	Umair Rashidi		a day ago on	Google	LA Hotel Metro exceeded all expectations! Impe...

```
df.isna().sum()
```

```
Name        3
Rating      3
Date        3
Platform    3
Review      3
dtype: int64
```

```
df.dropna(inplace=True)
```

Topic Modeling

```
from gensim.corpora import Dictionary
from gensim.models import LdaModel
from gensim.parsing.preprocessing import preprocess_string
```

```
# Preprocess the 'Review' column
processed_reviews = df['Review'].apply(preprocess_string)

# Create a dictionary representation of the documents
dictionary = Dictionary(processed_reviews)

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

# Convert the documents into bag-of-words (BoW) format
corpus = [dictionary.doc2bow(doc) for doc in processed_reviews]

# Train the LDA model
lda_model = LdaModel(corpus=corpus,
                     id2word=dictionary,
                     num_topics=5,
                     passes=10)

# Print the topics and their top words
topics = lda_model.print_topics(num_words=5)
for idx, topic in topics:
    print(f"Topic {idx}:")
    words = [word.split('*')[1].strip().strip('"') for word in topic.split('+')]
    print(', '.join(words))
    print()
```

```
Topic 0:
want, best, good, amaz, famili

Topic 1:
comfort, staff, metro, friendli, locat

Topic 2:
clean, comfort, staff, nice, provid

Topic 3:
staff, great, nice, experi, love

Topic 4:
good, servic, staff, food, experi
```

Sentiment Analysis

```
from textblob import TextBlob

# Calculate sentiment scores for each review
sentiment_scores = df['Review'].apply(lambda x: TextBlob(x).sentiment.polarity)

# Classify reviews as positive, negative, or neutral
positive_reviews = sum(score > 0 for score in sentiment_scores)
negative_reviews = sum(score < 0 for score in sentiment_scores)
neutral_reviews = sum(score == 0 for score in sentiment_scores)

# Calculate the average sentiment score
average_sentiment_score = sum(sentiment_scores) / len(sentiment_scores)

print(f"Number of positive reviews: {positive_reviews}")
print(f"Number of negative reviews: {negative_reviews}")
print(f"Number of neutral reviews: {neutral_reviews}")
print(f"Average sentiment score: {average_sentiment_score:.2f}")
```

```
Number of positive reviews: 114
Number of negative reviews: 6
Number of neutral reviews: 0
Average sentiment score: 0.47
```