```
# Importing necessary libraries
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from gensim import corpora
from gensim.models import LdaModel
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Download NLTK data
nltk.download('stopwords')
nltk.download('wordnet')

# Step 1: Load the scraped data
# Replace 'your_reviews_file.csv' with the name of your CSV
file.
#df = pd.read_csv('google (1).csv')
df = pd.read_csv(r'F:\APL\file_operation_node\SMA
\dominos_reviews_with_topics.csv')

# Replace 'reviews_column' with the name of the column that
holds the reviews
reviews = df['d4r55']

# Step 2: Preprocessing function for text cleaning
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def preprocess_text(text):
    # Remove punctuation, special characters, and convert to
lowercase
    text = re.sub(r'\W', ' ', text.lower())
    # Tokenize the text
    tokens = text.split()
    # Remove stopwords and apply lemmatization
    tokens = [lemmatizer.lemmatize(word) for word in tokens if
word not in stop_words]
    return ' '.join(tokens)

# Apply preprocessing to the reviews
df['cleaned_reviews'] = df['d4r55'].apply(preprocess_text)

# Step 3: Tokenizing the cleaned reviews
tokenized_reviews = [review.split() for review in
df['cleaned_reviews']]

# Step 4: Create a dictionary (vocabulary) and bag-of-words
representation of the reviews
dictionary = corpora.Dictionary(tokenized_reviews)
bow_corpus = [dictionary.doc2bow(review) for review in
tokenized_reviews]

# Step 5: Apply LDA for topic modeling
# Choosing the number of topics
num_topics = 5
```

```
lda_model = LdaModel(bow_corpus, num_topics=num_topics,
id2word=dictionary, passes=15)

# Step 6: Display the topics
for idx, topic in lda_model.print_topics(-1):
    print(f'Topic {idx}: {topic}')

# Step 7: Visualizing Topics with Word Cloud
for i, topic in lda_model.show_topics(formatted=False,
num_words=10):
    words = dict(topic)
    wordcloud = WordCloud(width=400, height=400,
background_color='white').generate_from_frequencies(words)

    # Plot the Word Cloud
    plt.figure()
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title(f"Topic {i}")
    plt.show()

# Step 8: Get the topic distribution for each review
print("\nTopic Distribution for each Review:")
for index, review in enumerate(bow_corpus):
    topic_distribution = lda_model.get_document_topics(review)
    print(f"Review {index}: {topic_distribution}")

# Optional: Saving the results to a CSV for further analysis
df['topic_distribution'] = [lda_model.get_document_topics(bow)
for bow in bow_corpus]
df.to_csv('dominos_reviews_with_topics.csv', index=False)
```