PHASE 3:

BUILDING THE MODEL AND DEVELOPING PREPROCESSING STEPS

TOPIC:

SENTIMENT ANALYSIS FOR MARKETING

STEP 1 - IMPORT LIBRARIES AND LOAD DATASET

First, we'll import the necessary libraries for text analysis and sentiment analysis, such as pandas for data handling, nltk for natural language processing, and Sentiment Intensity Analyzer for sentiment analysis.

We'll then download all of the NLTK corpus (a collection of linguistic data) using nltk.download().

Once the environment is set up, we will load a dataset of Amazon reviews using pd.read_csv (). This will create a Data Frame object in Python that we can use to analyze the data. We'll display the contents of the Data Frame using df.

CODE:

```
# import libraries

import pandas as pd

import nltk

from nltk.sentiment.vader import SentimentIntensityAnalyzer

from nltk.corpus import stopwords

from nltk.tokenize import word_tokenize

from nltk.stem import WordNetLemmatizer

# download nltk corpus (first time only)

import nltk

nltk.download('all')
```

```
# Load the amazon review dataset
df=pd.read_csv('https://raw.githubusercontent.com/pycaret/pycaret/master/datasets/amazon.csv')
df
```

OUTPUT:

	reviewText	Positive
0	This is a one of the best apps acording to a b...	1
1	This is a pretty good version of the game for ...	1
2	this is a really cool game. there are a bunch ...	1
3	This is a silly game and can be frustrating, b...	1
4	This is a terrific game on any pad. Hrs of fun...	1
...	...	...
19995	this app is fricken stupid.it froze on the kin...	0
19996	Please add me!!!!! I need neighbors! Ginger101...	1
19997	love it! this game. is awesome. wish it had m...	1
19998	I love love love this app on my side of fashio...	1
19999	This game is a rip off. Here is a list of thin...	0

STEP 2 - PREPROCESS TEXT

Let's create a function preprocess_text in which we first tokenize the documents using word_tokenize function from NLTK, then we remove step words using stepwords module from NLTK and finally, we lemmatize the filtered_tokens using WordNetLemmatizer from NLTK.

CODE:

```
# create preprocess_text function
def preprocess_text(text):
    # Tokenize the text
    tokens = word_tokenize(text.lower())
```

```
    # Remove stop words
    filtered_tokens = [token for token in tokens if token not in stopwords.words ('english')]
    # Lemmatize the tokens
    lemmatizer = WordNetLemmatizer()
    lemmatized_tokens = [lemmatizer.lemmatize(token) for token in filtered_tokens]
    # Join the tokens back into a string
    processed_text = ' '.join(lemmatized_tokens)
    return processed_text
# apply the function df
df['reviewText'] = df['reviewText'].apply(preprocess_text)
df
```

OUTPUT:

	reviewText	Positive
0	one best apps acording bunch people agree bomb...	1
1	pretty good version game free . lot different ...	1
2	really cool game . bunch level find golden egg...	1
3	silly game frustrating , lot fun definitely re...	1
4	terrific game pad . hr fun . grandkids love	1
...	...	...
19995	app fricken stupid.it froze kindle wont allow ...	0
19996	please add ! ! ! ! ! need neighbor ! ginger101...	1
19997	love ! game . awesome . wish free stuff house ...	1
19998	love love love app side fashion story fight wo...	1
19999	game rip . list thing make better & bull ; fir...	0

STEP 3 - NLTK SENTIMENT ANALYZER

He Natural Language Toolkit (NLTK) Sentiment Analyzer is a powerful tool in the field of natural language processing, designed to analyze and categorize textual data based on the sentiments expressed within it. NLTK, a widely used Python library, provides a comprehensive suite of libraries and programs for natural language processing tasks, including sentiment analysis.

The Sentiment Analyzer within NLTK utilizes machine learning algorithms and lexical resources to determine the sentiment polarity of a given text, classifying it as positive, negative, or neutral.

CODE:

```
# initialize NLTK sentiment analyzer
analyzer = SentimentIntensityAnalyzer()
# create get_sentiment function
def get_sentiment(text):
    scores = analyzer.polarity_scores(text)
    sentiment = 1 if scores['pos'] > 0 else 0
    return sentiment
# apply get_sentiment function
df['sentiment'] = df['reviewText'].apply(get_sentiment)
df
```

OUTPUT:

	reviewText	Positive	sentiment
0	one best apps acording bunch people agree bomb...	1	1
1	pretty good version game free . lot different ...	1	1
2	really cool game . bunch level find golden egg...	1	1
3	silly game frustrating , lot fun definitely re...	1	1
4	terrific game pad . hr fun . grandkids love	1	1
...	...	...	...
19995	app fricken stupid.it froze kindle wont allow ...	0	0
19996	please add ! ! ! ! ! need neighbor ! ginger101...	1	1
19997	love ! game . awesome . wish free stuff house ...	1	1
19998	love love love app side fashion story fight wo...	1	1
19999	game rip . list thing make better & bull ; fir...	0	1

The NLTK sentiment analyzer returns a score between -1 and + We have used a cut-off threshold of 0 in the get_sentiment function above. Anything above 0 is classified as 1 (meaning positive). Since we have actual labels, we can evaluate the performance of this method by building a confusion matrix.

CODE:

```
from sklearn.metrics import confusion_matrix
print(confusion_matrix(df['Positive'], df['sentiment']))
```

OUTPUT:

[[1131 3636] [576 14657]]

WE CAN ALSO CHECK THE CLASSIFICATION REPORT:

CODE:

```
from sklearn.metrics import classification_report
print (classification_report(df['Positive'], df['sentiment']))
```

OUTPUT:

	precision	recall	f1-score	support
0	0.66	0.24	0.35	4767
1	0.80	0.96	0.87	15233
accuracy			0.79	20000
macro avg	0.73	0.60	0.61	20000
weighted avg	0.77	0.79	0.75	20000

CONCLUSION:

The NLTK Sentiment Analyzer is a versatile and valuable tool in the realm of natural language processing. With its ability to automatically categorize text into positive, negative, or neutral sentiments, it empowers researchers, businesses, and organizations to gain insights from textual data, enabling them to make data-driven decisions, assess public opinion, and respond effectively to customer feedback.

Its utilization of machine learning algorithms and lexical resources makes it a robust choice for sentiment analysis tasks, and its integration within the broader NLTK library provides a wealth of other natural language processing capabilities. As a fundamental component of the NLTK toolkit, the Sentiment Analyzer continues to play a vital role in the field of text analysis, contributing to a better understanding of human sentiment and communication in the digital age.