PROBLEM DEFINITION AND DESIGN THINKING OF SENTIMENT ANALYSIS FOR MARKETING

PROBLEM STATEMENT:

The problem definition for sentiment analysis in marketing involves the systematic examination of customer sentiments, opinions, and emotional expressions from a variety of sources such as social media, reviews, surveys, and customer interactions. This analytical endeavour aims to extract valuable insights into how customers perceive and feel about products, services, brands, or marketing campaigns. These insights serve as a foundation for data-driven marketing decisions, enabling marketing professionals to optimize product strategies, fine-tune advertising campaigns, and enhance overall customer experiences. Furthermore, the project encompasses several challenges including handling diverse data types, processing large volumes of data efficiently, ensuring data quality through preprocessing, addressing linguistic diversity and multilingual support, achieving real-time analysis capabilities, selecting appropriate machine learning models, creating high-quality labeled datasets for supervised learning, and navigating ethical considerations and privacy concerns related to customer data, all while striving to reduce bias in sentiment analysis results and ensuring compliance with data protection regulations. The ultimate goal is to empower organizations to respond effectively to customer sentiment, improve marketing strategies, measure campaign success, identify opportunities for product enhancements, and proactively manage brand reputation.

GOALS:

The project goals for a sentiment analysis project in marketing are as follows:

1. **Customer Insight:** Gain a deep understanding of customer sentiments, opinions, and emotions to identify what factors trigger positive or negative reactions.

2**. Informed Marketing Strategies**: Equip marketing professionals with data-driven insights to refine product offerings, tailor advertising strategies, and optimize customer engagement tactics.

3. **Campaign Success Measurement:** Enable the measurement of the effectiveness of marketing campaigns by monitoring shifts in sentiment before, during, and after campaign launches.

4. **Product Improvement**: Identify areas for product or service enhancements based on customer feedback and sentiments, contributing to ongoing product development.

5**. Reputation Management**: Detect and address negative sentiment promptly to safeguard and manage brand reputation in the digital age.

DESIGN THINKING APPROACH:

1. Empathize: Understand User Needs

- Conduct interviews, surveys, and observations to deeply understand the needs and pain points of marketing professionals and stakeholders who will use sentiment analysis insights.

- Explore their challenges in decision-making, campaign effectiveness, and brand management.

2. Define: Frame the Problem

- Clearly define the problem statement, considering the insights gained from the empathize stage.

- Create user personas representing different stakeholders and their specific goals and challenges in utilizing sentiment analysis.

3. Ideate: Generate Solutions

- Organize brainstorming sessions with a cross-functional team to generate a wide range of ideas for addressing the defined problem.

- Encourage innovative thinking by considering different data sources, models, and visualization techniques.

4. Prototype: Build a Solution

- Develop a prototype or proof of concept for the sentiment analysis system, including selecting data sources, choosing initial models, and designing preliminary visualizations.

- Keep the prototype flexible to accommodate changes based on feedback.

5. Test: Gather Feedback

- Present the prototype to users, including marketing professionals, and collect their feedback on usability, accuracy, and usefulness.

- Use this feedback to refine the prototype and iterate on the design.

6. Implement: Deploy the Solution

- Once the sentiment analysis system has been refined and tested, proceed with full implementation.

- Ensure it integrates seamlessly with existing marketing analytics tools and workflows.

7. Iterate: Continuous Improvement

- Establish a feedback loop for ongoing improvement, allowing users to provide input on the system's performance and features.

- Continuously update sentiment analysis models based on evolving customer needs and market dynamics.

8. Monitor and Scale: Maintain Performance

- Implement real-time monitoring to ensure the sentiment analysis system is delivering accurate and up-to-date insights.

- Design the system to scale as the volume of data and user demands grow.

9. Ethical Considerations: Address Ethical Concerns

- Continuously evaluate and address ethical concerns related to data privacy, bias, and fairness in sentiment analysis.

- Ensure compliance with data protection regulations (e.g., GDPR, CCPA).

10. Training and Collaboration: Empower the Team

- Provide training and support to team members involved in using the sentiment analysis system.

- Foster collaboration between data scientists, analysts, marketers, and IT professionals throughout the project.

11. Communication: Share Insights Effectively

- Develop a clear communication strategy to share sentiment insights with stakeholders, including regular reports or dashboards.

- Ensure that the insights are presented in a format that is understandable and actionable for non-technical users.

PHASES OF DEVELOPMENT:

1. Data collection,
2. Data preprocessing,
3. Model development,
4. Model training, and
5. Model evaluation

1)DATASET USED:

- ✓ This dataset primarily contains text data in the form of tweets posted on Twitter.

- ✓ The dataset includes the following features or columns:
  - **`tweet_id`:** A unique identifier for each tweet.
  - **`airline_sentiment`**: The sentiment expressed in the tweet (positive, negative, or neutral).
  - **`airline_sentiment_confidence`**: The confidence level associated with the sentiment classification.
  - **`negativereason`**: The reason for negative sentiment if applicable.
  - `negativereason_confidence`: The confidence level associated with the negative reason classification.
  - **`airline`**: The name of the airline mentioned in the tweet.
  - **`airline_sentiment_gold`**: Additional information related to sentiment (e.g., if it's gold labeled data).
  - **`name`**: The Twitter handle of the user who posted the tweet.
  - **`retweet_count`**: The number of times the tweet was retweeted.
  - **`text`**: The text content of the tweet.

- ✓ The data is structured in a tabular format with rows and columns, which is common for structured data. Each row represents a tweet, and each column represents a specific attribute of the tweet.
- ✓ The `airline_sentiment` column serves as the target variable, which contains the sentiment labels (positive, negative, or neutral) for the tweets. This column is often used for sentiment classification tasks.
- ✓ The dataset contains 14,640 records or data points, with each row representing a tweet.
- ✓ These details provide a comprehensive understanding of the dataset's content, structure, and attributes, which is essential for anyone working with the data for analysis or machine learning tasks.

2)PREPROCESSING:

1. Data Loading and Inspection:

- The code begins by importing necessary libraries such as pandas, numpy, and scikit-learn.
- It loads a dataset from a CSV file named 'Tweets.csv' and prints the first few rows of the dataset for inspection.

2. Text Cleaning:

- This is the process of removing or correcting any noise, irrelevant characters, or artifacts in the text data. Common text cleaning operations include:
  - Removing HTML tags, if the text contains web content.
  - Handling special characters, punctuation, and non-alphanumeric characters.
  - Converting text to lowercase to ensure uniformity.
  - Handling whitespace and line breaks
- The text data in the 'text' column of the dataset is pre-processed.
- Text is converted to lowercase.
- Non-alphabetic characters are removed from the text using regex.
- The cleaned text is printed for reference.

3. Tokenization:

- Tokenization is the process of breaking down the text into smaller units called tokens.
- These tokens can be words, phrases, or even individual characters, depending on the specific task. Tokenization is a crucial step for text analysis and natural language processing (NLP)
- . It splits the text into meaningful units for further analysis
- The NLTK library is used to tokenize the cleaned text data.
- The tokenized data is stored in a new column 'tokens'.
- The tokenized data is printed for reference.

4. Stop Words Removal:

- Stop words from the NLTK library are used to filter out common English stop words from the tokenized text.
- Stop words are common words (e.g., "the," "and," "is") that are often removed from text data because they don't typically carry meaningful information for many NLP tasks.
- The resulting tokenized data with stop words removed is stored in the 'tokens' column.
- The modified data is printed for reference.

5. Label Encoding:

- Label encoding is a common preprocessing technique used when dealing with categorical data in machine learning and data analysis.
- It involves converting categorical values into numerical values to make them compatible with machine learning algorithms that expect numeric inputs.
- The 'airline_sentiment' column is mapped from categorical labels ('negative', 'neutral', 'positive') to numeric labels (0, 1, 2).
- The label-encoded 'airline_sentiment' is printed for reference.

5.Stratified Sampling:

- Stratified sampling is a sampling technique commonly used in statistics and research to ensure that a representative sample is taken from a population.
- It is particularly useful when you have a diverse population with different subgroups, and you want to ensure that each subgroup is adequately represented in the sample

6. Splitting Data:

- The data is split into training and testing sets using Stratified Sampling.
- The split is performed with an 80-20 ratio, ensuring that the distribution of labels in the training and testing sets is similar.
- The split data is printed for reference.

3) FEATURE EXTRACTION TECHNIQUE USED:

TF-IDF (Term Frequency-Inverse Document Frequency):

- **Term Frequency (TF):** It measures the frequency of a term (word) within a document. It calculates how often a word occurs in a specific document relative to the total number of words in that document.
- **Inverse Document Frequency (IDF):** It quantifies how unique or rare a term is across the entire dataset (corpus). Words that appear in many documents have lower IDF scores, while words that are specific to a few documents have higher IDF scores.
- **TF-IDF Score:** It combines TF and IDF to calculate a numerical score for each term in each document. The TF-IDF score is higher for terms that are frequent in a specific document but rare in the entire corpus.
- The TF-IDF feature extraction technique is widely used in text classification tasks like sentiment analysis, document classification, and information retrieval. It helps to convert text data into a numerical format that machine learning models can process, while also giving more weight to informative words and reducing the impact of common, less informative words.

Machine Learning Algorithm:

Logistic Regression:

The chosen algorithm for sentiment analysis in this code is logistic regression. Logistic regression is a widely used algorithm for binary and multi-class classification tasks, and it's suitable for text classification problems like sentiment analysis. It's a linear model that can model the relationship between the features (TF-IDF representations of text) and the target variable (sentiment labels) effectively.

Model Training:

1.Data Preprocessing:

The code performs several data preprocessing steps, including text cleaning, tokenization, and stop word removal, to prepare the text data for analysis. These steps help clean and structure the text data.

2.Stratified Sampling:

Stratified sampling is used to split the dataset into training and test sets. This ensures that the distribution of sentiment labels in both sets is similar to the original dataset, which is essential to maintain the representation of different sentiment classes in the training and testing data.

3.TF-IDF Vectorization:

The text data is transformed into numerical features using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. TF-IDF is a technique to represent the importance of words in text data, and it's commonly used in text-based machine learning tasks.

4.Model Training:

A logistic regression model is trained using the TF-IDF-transformed training data. The model learns the relationship between the text features and the sentiment labels during training.

Evaluation Metrics:

Classification Report:

The code evaluates the model's performance using the `classification_report` function from scikit-learn. This function provides various classification metrics for each class in the multi-class classification problem. The metrics typically included are precision, recall, F1-score, and support for each class. These metrics offer insights into how well the model performs for different sentiment categories (negative, neutral, positive).

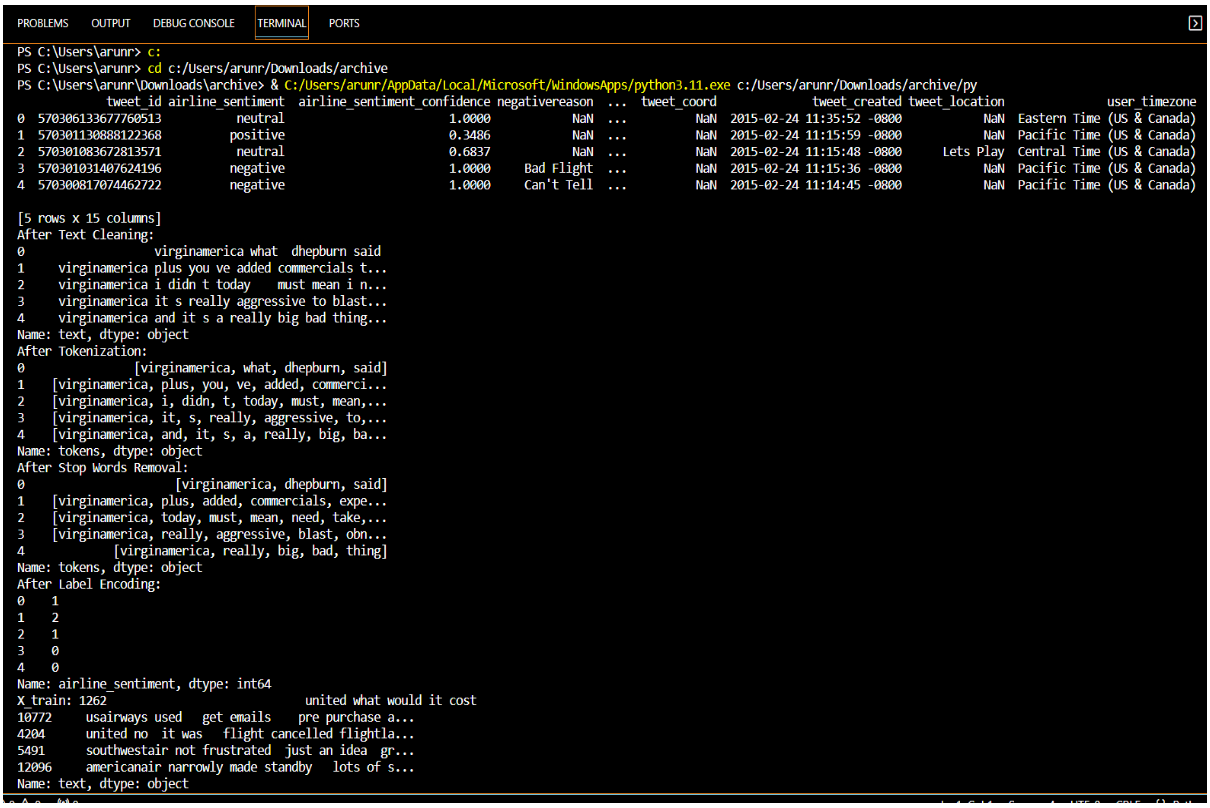
In summary, logistic regression is chosen as the classification algorithm, and it's trained on pre-processed text data with TF-IDF vectorization. The evaluation is performed using a classification report, which provides a comprehensive assessment of the model's performance across different sentiment classes. This approach is common for sentiment analysis tasks and can provide valuable insights into the model's effectiveness.

SOURCE CODE:

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.model_selection import StratifiedShuffleSplit
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords


#### Load your dataset, assuming it's in a CSV file
data = pd.read_csv('Tweets.csv')


#### Inspect the first few rows of the dataset
print(data.head())


#### Text cleaning
data['text'] = data['text'].str.lower()
data['text'] = data['text'].str.replace('[^a-zA-Z]', ' ', regex=True)
print("After Text Cleaning:")
print(data['text'].head())


#### Tokenization (using NLTK as an example)
data['tokens'] = data['text'].apply(word_tokenize)
print("After Tokenization:")
print(data['tokens'].head())
```

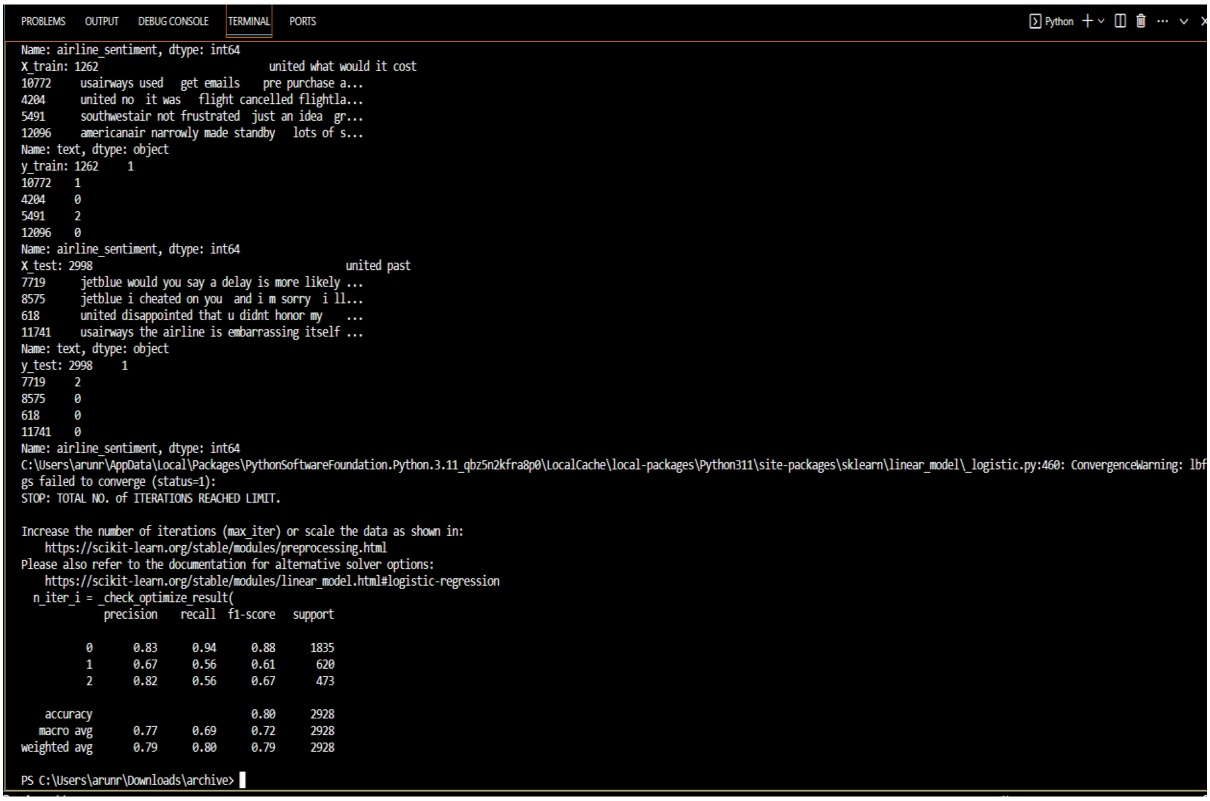
```
#### Remove stop words (using NLTK as an example)
stop_words = set(stopwords.words('english'))
data['tokens'] = data['tokens'].apply(lambda tokens: [word for word in tokens
if word not in stop_words])
print("After Stop Words Removal:")
print(data['tokens'].head())


#### Label encoding (assuming you have 'sentiment' as the label column)
sentiment_mapping = {'negative': 0, 'neutral': 1, 'positive': 2}
data['airline_sentiment'] = data['airline_sentiment'].map(sentiment_mapping)
print("After Label Encoding:")
print(data['airline_sentiment'].head())


X = data['text'] # Input features
y = data['airline_sentiment'] # Target variable


#### Perform Stratified Sampling with a 80-20 split
stratified_split = StratifiedShuffleSplit(n_splits=1, test_size=0.2,
random_state=42)
for train_index, test_index in stratified_split.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
print("X_train:", X_train.head())
print("y_train:", y_train.head())
print("X_test:", X_test.head())
print("y_test:", y_test.head())


#### Extract TF-IDF features
vectorizer = TfidfVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)


#### Train a logistic regression model
clf = LogisticRegression()
clf.fit(X_train_vectorized, y_train)


#### Evaluate the model on the test set
y_pred = clf.predict(X_test_vectorized)
print(classification_report(y_test, y_pred))
```

RESULT:



```
PS C:\Users\arunr> c:
PS C:\Users\arunr> cd c:/Users/arunr/Downloads/archive
PS C:\Users\arunr\Downloads\archive> & C:/Users/arunr/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/arunr/Downloads/archive/py
             tweet_id airline_sentiment  airline_sentiment_confidence negativereason  ...  tweet_coord              tweet_created tweet_location               user_timezone
0  570306133677760513           neutral                        1.0000            NaN  ...          NaN  2015-02-24 11:35:52 -0800            NaN  Eastern Time (US & Canada)
1  570301130888122368          positive                        0.3486            NaN  ...          NaN  2015-02-24 11:15:59 -0800            NaN  Pacific Time (US & Canada)
2  570301083672813571           neutral                        0.6837            NaN  ...          NaN  2015-02-24 11:15:48 -0800      Lets Play  Central Time (US & Canada)
3  570301031407624196          negative                        1.0000     Bad Flight  ...          NaN  2015-02-24 11:15:36 -0800            NaN  Pacific Time (US & Canada)
4  570300817074462722          negative                        1.0000     Can't Tell  ...          NaN  2015-02-24 11:14:45 -0800            NaN  Pacific Time (US & Canada)

[5 rows x 15 columns]
After Text Cleaning:
0                    virginamerica what  dhepburn said
1     virginamerica plus you ve added commercials t...
2     virginamerica i didn t today    must mean i n...
3     virginamerica it s really aggressive to blast...
4     virginamerica and it s a really big bad thing...
Name: text, dtype: object
After Tokenization:
0                 [virginamerica, what, dhepburn, said]
1    [virginamerica, plus, you, ve, added, commerci...
2    [virginamerica, i, didn, t, today, must, mean,...
3    [virginamerica, it, s, really, aggressive, to,...
4    [virginamerica, and, it, s, a, really, big, ba...
Name: tokens, dtype: object
After Stop Words Removal:
0                       [virginamerica, dhepburn, said]
1    [virginamerica, plus, added, commercials, expe...
2    [virginamerica, today, must, mean, need, take,...
3    [virginamerica, really, aggressive, blast, obn...
4              [virginamerica, really, big, bad, thing]
Name: tokens, dtype: object
After Label Encoding:
0    1
1    2
2    1
3    0
4    0
Name: airline_sentiment, dtype: int64
X_train: 1262                               united what would it cost
10772     usairways used   get emails    pre purchase a...
4204      united no  it was   flight cancelled flightla...
5491      southwestair not frustrated  just an idea  gr...
12096     americanair narrowly made standby   lots of s...
Name: text, dtype: object
```

+



```
Name: airline_sentiment, dtype: int64
X_train: 1262                               united what would it cost
10772     usairways used   get emails    pre purchase a...
4204      united no  it was   flight cancelled flightla...
5491      southwestair not frustrated  just an idea  gr...
12096     americanair narrowly made standby   lots of s...
Name: text, dtype: object
y_train: 1262     1
10772    1
4204     0
5491     2
12096    0
Name: airline_sentiment, dtype: int64
X_test: 2998                                       united past
7719      jetblue would you say a delay is more likely ...
8575      jetblue i cheated on you  and i m sorry  i ll...
618       united disappointed that u didnt honor my    ...
11741     usairways the airline is embarrassing itself ...
Name: text, dtype: object
y_test: 2998     1
7719     2
8575     0
618      0
11741    0
Name: airline_sentiment, dtype: int64
C:\Users\arunr\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\linear_model\_logistic.py:460: ConvergenceWarning: lbf
gs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
              precision    recall  f1-score   support

           0       0.83      0.94      0.88      1835
           1       0.67      0.56      0.61       620
           2       0.82      0.56      0.67       473

    accuracy                           0.80      2928
   macro avg       0.77      0.69      0.72      2928
weighted avg       0.79      0.80      0.79      2928

PS C:\Users\arunr\Downloads\archive> █
```