**TEAM MEMBERS**

SALOPRIYA P

AKSHAYA M N

PAVITHRA V

HEMAPRIYA V

**Project Module: Training the Model and Evaluating its**

**Performance**

**PHASE 4**

**Abstract:**

The project aims to develop a Sentiment Analysis for Marketing using Amazon Product Reviews Datasets to provide exceptional customer service and support on a website or application. This project module document outlines the introduction, problem definition, needs, software and hardware requirements, step-by-step methods, and a final conclusion for the project.

**Step-by-Step Methods:**

**1.Problem Definition :**

Understand the problem and user needs.

**2.Data Collection:**

Gather data from various sources like social media (Twitter, Facebook, Instagram), online reviews or customer surveys. Consider using web scraping tools or APIs.

**3.Data Preprocessing:**

Clean and preprocess the text data. This involves tasks like removing special characters, converting text to lowercase, and tokenization.

**4.Labeling:**

For a supervised approach, label your data. Assign sentiment labels (positive, negative, neutral) to your dataset. You can use crowdsourcing platforms or sentiment analysis tools to help with this.

**5.Feature Extraction:**

Convert text data into numerical features. Common techniques include TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings (Word2Vec, GloVe).

**6.Model Selection:**

Choose a sentiment analysis model. Common choices include Naïve Bayes, Support Vector Machines or more advanced methods like deep learning with Recurrent Neutral Networks (RNNs) or Transformers (e.g.,BERT).

**7.Model Training:**

Train your selected model on the labeled dataset. Use a portion of the data for training and reserve some for validation and testing.

**8.Model Evaluation:**

Assess the model’s performance using metrics like accuracy, precision, recall, and F1-score. Adjust the model if necessary.

**9.Sentiment Analysis:**

Apply the trained model to analyze new text data. This could be real-time social media data or any text relevant to your marketing objectives.

**10.Visualization:**

Create visualizations to present the sentiment analysis results. Word clouds, bar charts, and sentiment over time plots can be useful.

**11.Actionable Insights:**

Interpret the results to gain insights. Identify trends, popular topics, and areas where sentiment is strongly positive or negative.

**12.Feedback Loop:**

Continuously monitor sentiment and adapt your marketing strategies accordingly. Use the feedback to improve customer satisfaction and brand perception.

**13.Reports and Dashboards:**

Create reports and dashboards to share the sentiment analysis results with relevant stakeholders.

**14.Iterate and Improve:**

Regularly review and refine your sentiment analysis process to ensure its effectiveness in guiding marketing strategies.

**Steps for implementation:**

**Step 1: Import the necessary Libraries**

Import the necessary libraries. For working with Kaggle datasets, you need the Kaggle API library.

Python

import re

import random

import pandas as pd

import kaggle

**Step 2: Download and Load a Kaggle Dataset**

To use a Kaggle dataset, you first need to download it. Make sure you have the Kaggle API credentials set up, and then download a suitable dataset. For this example, we will use a simple CSV file. You can request it with any other Kaggle dataset you prefer.

**Python**

**#Download the dataset from Kaggle**

Kaggle.api.authenticate(api\_key=’YOUR\_API\_KEY’)

Kaggle.api.dataset\_download\_files(‘Kaggle/dataset-name’, path=’/’, unzip=True)

**#Load the dataset**

data=pd.read\_csv(‘Twitter-airline-sentiment-dataset.csv’)

**Step 3: Preprocess the Dataset**

Preprocess the Kaggle dataset to amazon products for the sentiment analysis for marketing.

**Python**

import pandas as pd

import re

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

from nltk.stem import WordNetLemmatizer

**#Sample dataset**

data = [

{“tweet\_id”: 570306133677760513, “sentiment”: “Neutral”, “confidence\_level”: 1.0,”text”: ”cairdin”},

{“tweet\_id”: 570301130888122368, “sentiment”: “Positive”, “confidence\_level”: 0.3,”text”: “jnardino”},

{“tweet\_id”: 570301083672813571, “sentiment”: “Neutral”, “confidence\_level”: 0.6,”text”:“yvonalyn”},

{“tweet\_id”: 570301031407624196, “sentiment”: “Negative”, “confidence\_level”: 1.0,”text”: “Bad Flight”}

]

**#Convert the data to a pandas DataFrame**

**df = pd.DataFrame(data)**

**#Convert text to lowercase**

df[‘text’] = df[‘text’].str.lower()

**#Remove special characters and digits**

df[‘text’] = df[‘text’].apply(lambda x: re.sub(r’[^\w\s]’, ‘ ‘, x))

df[‘text’] = df[‘text’].apply(lambda x: re.sub(r”\d+”, “ “, x))

**#Tokenize text**

df[‘text’] = df[‘text’].apply(word\_tokenize)

**#Remove stop words**

stop\_words = set(stopwrods.words(‘english’))

df[‘text’] = df[‘text’].apply(lambda x: [word for word in a if word not in stop\_words])

**#Lemmatization**

Lemmatizer = WordNetLemmatizer()

df[‘text’] = df[‘text’].apply(lambda x: [lemmatizer.lemmatize(word) for word in x])\

**Step 4:** **Splitting the dataset**

**Python**

**#split the data into training and testing sets**

X = df[‘text’]

Y = df[‘sentiment’]

X\_train. X\_text, y\_train, y\_test = train\_text\_split(X, y, text\_size=0.2, random\_state=42)

**#Vectorize the text data**

Vectorizer = TfidfVectorizer()

X\_train\_vect = vectorizer.fit\_transform(X\_train)

X\_text\_vectt = vectorizer.transform(X\_test)

**Step 5: Training the model**

**#Train a Support Vector Machine (SVM) model**

svm\_model = SVM(kernel=’linear’)

svm\_model.fit(X\_train\_vect, y\_train)

**#Predict the sentiment labels for the test set**

y\_pred = svm\_model.predict(X\_test\_vect)

**Step 6: Evaluating its Performance**

**#Evaluate the model**

accuracy = accuracy\_score(y\_test, y\_pred)

precision, recall, f1\_score, \_ = precision\_recall\_fscore\_support(y\_test, y\_pred, average=’weighted’)

**#Print the evaluation metrics**

print(“Accuracy:”, accuracy)

print(“Precision:”, precision)

print(“Recall:”, recall)

print(“F1 Score:”, f1\_score)

**#Print the classification report and confusion matrix**

print(“\nClassification Report:\n”, classification\_report(y\_test, y\_pred))

print(“\nConfusion Matrix:\n”, confusion\_matrix(y\_test, y\_pred))

**Conclusion:**

In this implementation, based on the process and evaluation performed, the Support Vector Machine (SVM) model demonstrates a reasonable performance in sentiment analysis on the given dataset. The evaluation metrics indicates a decent accuracy, precision, recall, and F1-score.