**SENTIMENT ANALYSIS ON PRODUCT REVIEW**

**Source Code**

***Submitted in partial fulfillment for the award of the degree of***

**Master of Computer Applications Batch**

**July - 2023**

***Submitted to***



**CENTRE FOR DISTANCE & ONLINE EDUCATION CHANDIGARH UNIVERSITY**

**Submitted by**

**Name: Shivam Rai**

**Enrolment No: O23MCA110089**

**Subject Code : 23ONMCR-753**

**Source Code**

**app.py**

import pandas as pd

import nltk

import matplotlib.pyplot as plt

import seaborn as sns

from wordcloud import WordCloud

from nltk.corpus import stopwords

from sklearn.model\_selection import train\_test\_split

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, confusion\_matrix

import streamlit as st

import os

# Download NLTK stopwords

nltk.download('stopwords')

stop\_words = set(stopwords.words('english'))

# CSV file path

file\_path = 'flipkart\_data.csv'

# Preprocessing function

def preprocess\_reviews\_stopwords(df):

    df['review'] = df['review'].astype(str).str.lower()

    df['review'] = df['review'].apply(lambda x: ' '.join([word for word in x.split() if word not in stop\_words]))

    df['sentiment'] = df['rating'].apply(lambda x: 1 if x >= 4 else 0)

    return df

# Load and preprocess data

if os.path.exists(file\_path):

    df = pd.read\_csv(file\_path)

    df\_cleaned = preprocess\_reviews\_stopwords(df)

else:

    df\_cleaned = pd.DataFrame(columns=['review', 'rating', 'sentiment'])

# Initialize variables for modeling

model = None

accuracy = None

conf\_matrix = None

labels = None

# Train model only if enough data

if not df\_cleaned.empty and len(df\_cleaned) > 5:

    tfidf = TfidfVectorizer()

    X = tfidf.fit\_transform(df\_cleaned['review'])

    y = df\_cleaned['sentiment']

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(

        X, y, stratify=y, test\_size=0.2, random\_state=42

    )

    model = DecisionTreeClassifier(random\_state=42)

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

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

    labels = model.classes\_

    conf\_matrix = confusion\_matrix(y\_test, y\_pred, labels=labels)

# --- Streamlit UI ---

st.set\_page\_config(page\_title="Sentiment App", layout="wide")

st.title("🛒 Sentiment Product Analysis Review System")

st.markdown(f"\*\*📊 Total Reviews in Dataset:\*\* {len(df\_cleaned)}")

# Show full dataset

st.subheader("📄 Full Review Dataset")

st.dataframe(df\_cleaned[['review', 'rating', 'sentiment']])

if not df\_cleaned.empty and model is not None:

    st.subheader("Sentiment Distribution")

    sentiment\_counts = df\_cleaned['sentiment'].value\_counts().sort\_index()

    sentiment\_labels = ['Negative', 'Positive']

    fig, ax = plt.subplots()

    sns.barplot(x=sentiment\_labels, y=sentiment\_counts.values, palette='coolwarm', ax=ax)

    ax.set\_xlabel("Sentiment")

    ax.set\_ylabel("Count")

    st.pyplot(fig)

    st.subheader("Word Cloud for Positive Reviews")

    positive\_text = ' '.join(df\_cleaned[df\_cleaned['sentiment'] == 1]['review'])

    wordcloud = WordCloud(width=800, height=400).generate(positive\_text)

    fig\_wc, ax\_wc = plt.subplots(figsize=(10, 4))

    ax\_wc.imshow(wordcloud, interpolation='bilinear')

    ax\_wc.axis('off')

    st.pyplot(fig\_wc)

    st.subheader("Model Accuracy and Confusion Matrix")

    st.write(f"\*\*Model Accuracy:\*\* {accuracy:.2f}")

    fig\_cm, ax\_cm = plt.subplots()

    sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap="Blues",

                xticklabels=labels, yticklabels=labels, ax=ax\_cm)

    ax\_cm.set\_xlabel("Predicted")

    ax\_cm.set\_ylabel("Actual")

    st.pyplot(fig\_cm)

else:

    st.warning("⚠️ Not enough data to train the model. Please add more reviews.")

# Add new review section

st.subheader("📩 Add a New Review")

with st.form("review\_form", clear\_on\_submit=True):

    review\_text = st.text\_area("Enter your review:")

    review\_rating = st.slider("Select Rating (1 to 5):", 1, 5, 3)

    submitted = st.form\_submit\_button("Add Review")

    if submitted and review\_text.strip() != "":

        new\_data = pd.DataFrame({

            'review': [review\_text],

            'rating': [review\_rating]

        })

        new\_data = preprocess\_reviews\_stopwords(new\_data)

        new\_data.to\_csv(file\_path, mode='a', index=False, header=not os.path.exists(file\_path))

        st.session\_state["review\_added"] = True

        st.rerun()

# ✅ Show success message if review was just added

if st.session\_state.get("review\_added", False):

    st.success("✅ Review added successfully.")

    st.session\_state["review\_added"] = False

st.markdown("---")

st.caption("Created with ❤️ using Streamlit")

**requirements.txt**

streamlit

pandas

nltk

scikit-learn

matplotlib

seaborn

wordcloud

**importnltk.py**

import nltk

nltk.download('punkt')

nltk.download('stopwords')