Hindi Sentiment Analysis

Umang Bid Computer Engineering Shah and Anchor Kutchhi Engineering college BE3-7	Anoushka Dighe Computer Engineering Shah and Anchor Kutchhi Engineering College BE3-16	Nimish Dhumale Computer Engineering Shah and Anchor Kutchhi Engineering college BE3-15

Shah and Anchor Kutchhi Engineering College

DEPARTMENT OF COMPUTER ENGINEERING
SHAH AND ANCHOR KUTCHHI ENGINEERING COLLEGE
CHEMBUR, MUMBAI – 400088.

2023 -2024

Problem Statement

Sentiment analysis, also known as opinion mining, is a natural language processing (NLP) task that involves determining the sentiment or emotion expressed in a piece of text. This task is crucial for understanding public opinion, customer feedback, and social media discussions. The focus of this problem statement is to perform sentiment analysis on Hindi text data, which is a critical task in understanding sentiment within the Hindi-speaking community.

Technologies Used

1. **Python:** The code is written in the Python programming language, which is widely used in data analysis, machine learning, and NLP tasks.
2. **Pandas:** The pandas library is used for data manipulation and analysis. It provides data structures like DataFrames, making it easier to work with structured data.
3. **Scikit-Learn (sklearn):** Scikit-Learn is a popular machine learning library in Python. In your code, you use several modules from Scikit-Learn, including:
4. **train_test_split:** This module is used to split the dataset into training and testing sets.
5. **TfidfVectorizer:** It is used to convert the text data into TF-IDF (Term Frequency-Inverse Document Frequency) features, which are essential for training a text classification model.
6. **SVC (Support Vector Classification):** This is the machine learning algorithm used for classification. In your code, you use an SVM classifier with a linear kernel.
7. accuracy_score: It is used to calculate the accuracy of the model.
8. **classification_report:** This module provides a comprehensive report on various classification metrics.
9. **Joblib:** The joblib library is used for model serialization and deserialization. It is used to save and load machine learning models and other objects efficiently.
10. **CSV Data:** Your code reads data from a CSV file ('emotions.csv') using Pandas. CSV (Comma-Separated Values) is a common format for storing structured data.
11. **Text Preprocessing:** Although not explicitly mentioned in your code, text preprocessing is an essential step in NLP. Preprocessing may include tasks like text cleaning, tokenization, and handling stopwords.
12. **Machine Learning Model:** Your code uses a Support Vector Machine (SVM) model for text classification. SVMs are a popular choice for binary and multi-class classification tasks, including sentiment analysis.
13. **TF-IDF:** The TF-IDF vectorization technique is used to convert text data into numerical features, which are then used as input for the SVM model. This technique is essential for transforming text data into a format that machine learning models can work with.
14. **Model Serialization:** The code saves the trained model and TF-IDF vectorizer to files using joblib for later use. This is a common practice in machine learning to avoid retraining the model every time.

Input Dataset

Link for the dataset :

https://www.kaggle.com/datasets/chiragmvarma/hindi-sentiment-analysis/data

Source code :

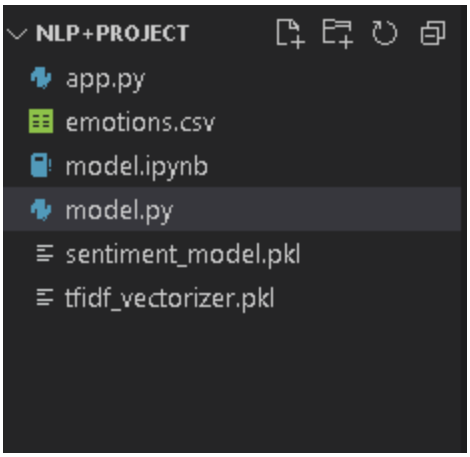
```
import joblib
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report


data = pd.read_csv('C:/Users/umang/Downloads/NLP+project -
Copy/NLP+project/emotions.csv')

X = data['Sentences']
y = data['Label']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

tfidf_vectorizer = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

```
model = SVC(kernel='linear', C=1.0, probability=True)
model.fit(X_train_tfidf, y_train)

y_pred = model.predict(X_test_tfidf)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")

report = classification_report(y_test, y_pred)
print("Classification Report:\n", report)



joblib.dump(model, 'sentiment_model.pkl')
joblib.dump(tfidf_vectorizer, 'tfidf_vectorizer.pkl')

loaded_model = joblib.load('sentiment_model.pkl')
loaded_vectorizer = joblib.load('tfidf_vectorizer.pkl')
new_text = ["This is amazing!", "Worst experience ever."]
new_text_tfidf = loaded_vectorizer.transform(new_text)
predictions = loaded_model.predict(new_text_tfidf)

print("Predictions:", predictions)
print("Model Accuracy:", accuracy)
print("Classification Report:\n", report)
```

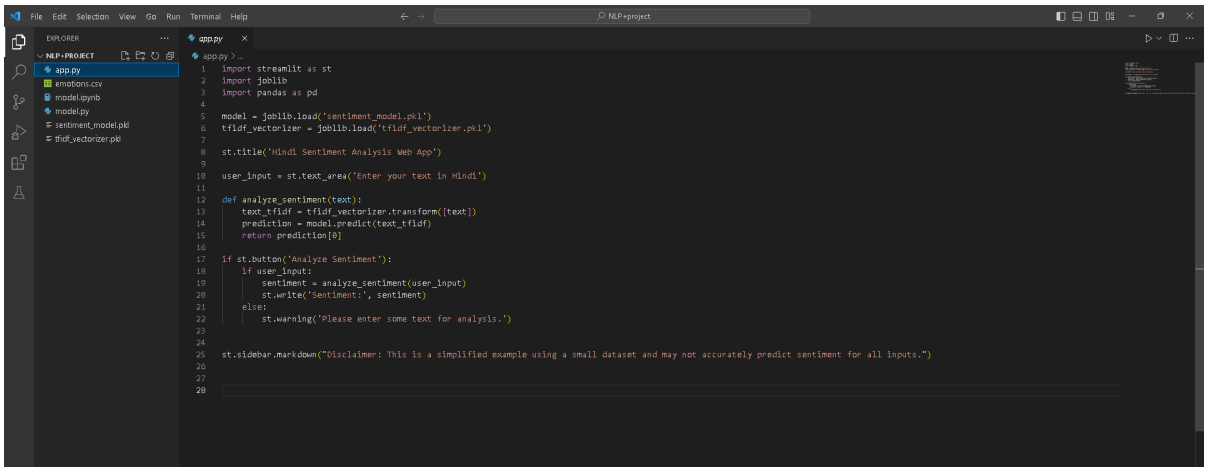
```
import streamlit as st
import joblib
import pandas as pd

model = joblib.load('sentiment_model.pkl')
tfidf_vectorizer = joblib.load('tfidf_vectorizer.pkl')

st.title('Hindi Sentiment Analysis Web App')

user_input = st.text_area('Enter your text in Hindi')

def analyze_sentiment(text):
    text_tfidf = tfidf_vectorizer.transform([text])
    prediction = model.predict(text_tfidf)
    return prediction[0]

if st.button('Analyze Sentiment'):
    if user_input:
        sentiment = analyze_sentiment(user_input)
        st.write('Sentiment:', sentiment)
    else:
        st.warning('Please enter some text for analysis.')


st.sidebar.markdown("Disclaimer: This is a simplified example using a
small dataset and may not accurately predict sentiment for all inputs.")
```

Screenshots

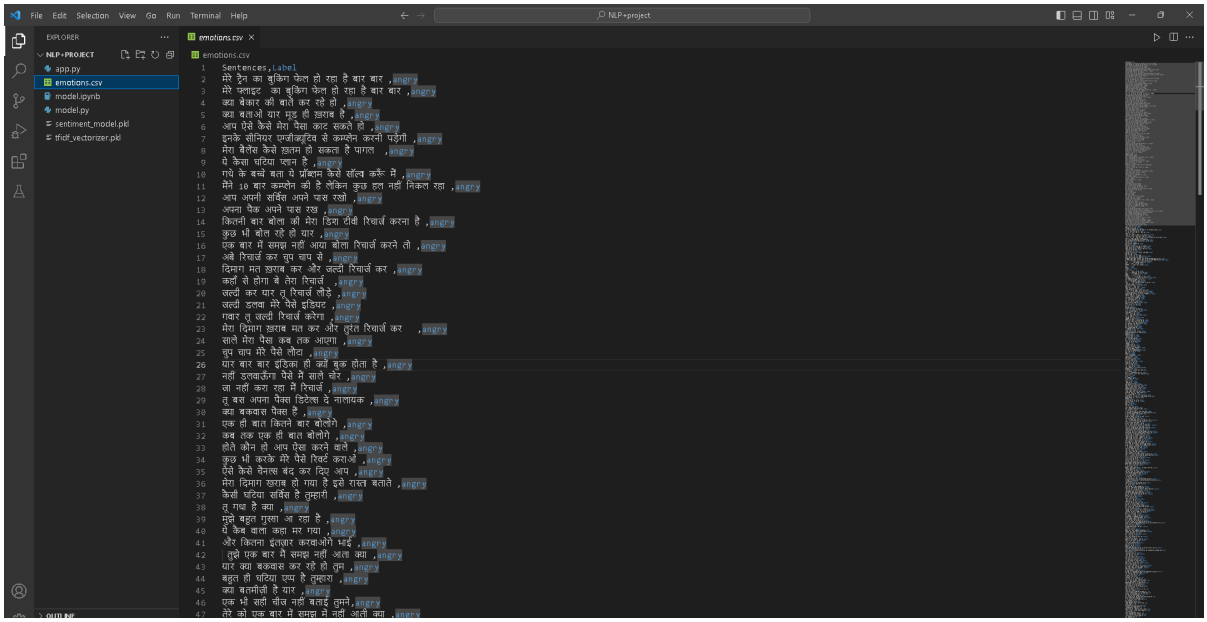
Structure of the project:



App.py file :

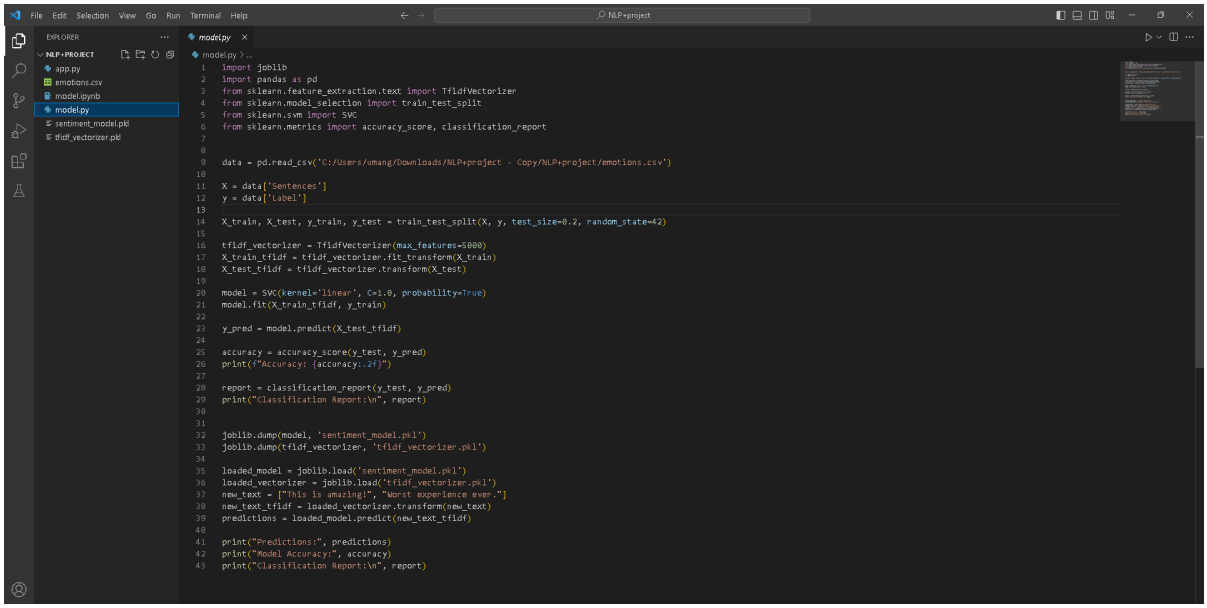


Emotions.csv file:



```
Sentences,Label
मेरे ट्रैन का बुकिंग फेल हो रहा है बार बार ,angry
मेरे फ्लाइट  का बुकिंग फेल हो रहा है बार बार ,angry
क्या बेकार की बातें कर रहे हो ,angry
क्या बताओ यार मूड ही ख़राब है ,angry
आप ऐसे कैसे मेरा पैसा काट सकते हो ,angry
इनके सीनियर पर्सोंअलिटी से कम्प्लेन करनी पड़ेगी ,angry
मेरा वैलेंस कैसे ख़तम हो सकता है पागल  ,angry
ये कैसा घटिया प्लान है ,angry
गधे के बच्चे बता ये प्रॉब्लम कैसे सॉल्व करूँ मैं ,angry
मैंने 10 बार कम्प्लेन की है लेकिन कुछ हल नहीं निकल रहा ,angry
आप अपनी सर्विस अपने पास रखो ,angry
अपना पैक अपने पास रख ,angry
कितनी बार बोला की मेरा दिन रीची रिचार्ज करना है ,angry
कुछ भी बोल रहे हो यार ,angry
एक बार में समझ नहीं आया बोला रिचार्ज करने ती ,angry
अबे रिचार्ज कर चुप चाप से ,angry
दिमाग मत ख़राब कर और जल्दी रिचार्ज कर ,angry
कहाँ से होगा बे तेरा रिचार्ज  ,angry
जल्दी कर पार तू रिचार्ज तोड़े ,angry
जल्दी डलवा मेरे पैसे इडियट ,angry
गवार तू जल्दी रिचार्ज करेगा ,angry
मेरा दिमाग ख़राब मत कर और तुरंत रिचार्ज कर  ,angry
सालै मेरा पैसा कब तक आएगा ,angry
चुप चाप मेरे पैसे लौटा ,angry
यार बार बार इंटिका ही क्यों चूक होता है ,angry
नहीं डलवाऊँगा पैसे ने चारो खोर ,angry
जो नहीं करा रहा मैं रिचार्ज ,angry
तू बस अपना पैक्स डिटेल्स दे नालायक ,angry
क्या बकवास पैक्स है ,angry
एक ही बात कितने बार बोलेंगे ,angry
कब तक एक ही बात बोलोगे ,angry
होते कौन हो आप ऐसा करने वाले ,angry
कुछ भी करके मेरे पैसे रिवर्ट करा ओ ,angry
ऐसे कैसे चैनल्स बंद कर दिए आप ,angry
मेरा दिमाग खराब हो गया है इसे रास्ता बताते ,angry
कैसी घटिया सर्विस है तुम्हारी ,angry
तू गाध है क्या ,angry
मुझे बहुत गुस्सा आ रहा है ,angry
ये फेक बाबा कहा मर गया ,angry
और कितना इंतज़ार करवाओगे भई ,angry
तुझे एक बार मै समझ नहीं आता क्या ,angry
यार क्या बकवास कर रहे हो तुम ,angry
बहुत ही घटिया एप्प है तुम्हारा ,angry
कब बतमीज़ी है यार ,angry
एक भी सही चीज नहीं बताई तुमने,angry
तेरे को एक बार में समझ में नहीं आती क्या ,angry
```

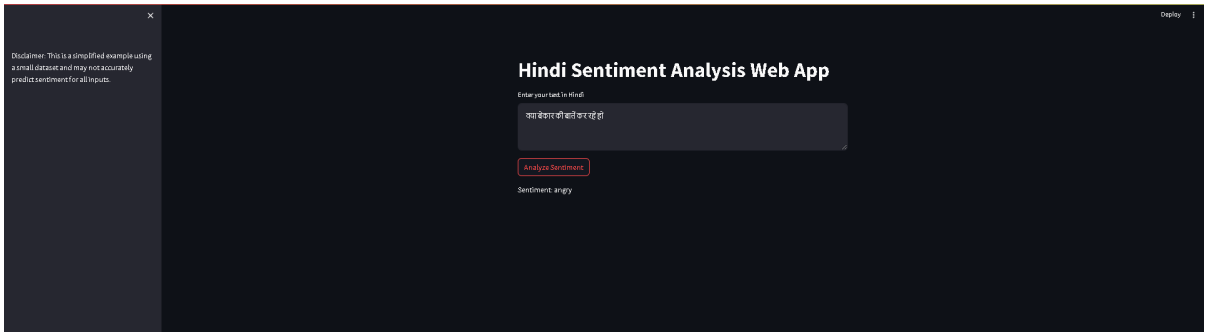
Model.py file :



```
import joblib
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report


data = pd.read_csv('C:/Users/umang/Downloads/NLP+project - Copy/NLP+project/emotions.csv')

X = data['Sentences']
y = data['Label']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

tfidf_vectorizer = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

model = SVC(kernel='linear', C=1.0, probability=True)
model.fit(X_train_tfidf, y_train)

y_pred = model.predict(X_test_tfidf)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")

report = classification_report(y_test, y_pred)
print("Classification Report:\n", report)


joblib.dump(model, 'sentiment_model.pkl')
joblib.dump(tfidf_vectorizer, 'tfidf_vectorizer.pkl')

loaded_model = joblib.load('sentiment_model.pkl')
loaded_vectorizer = joblib.load('tfidf_vectorizer.pkl')
new_text = ["This is amazing!", "Worst experience ever."]
new_text_tfidf = loaded_vectorizer.transform(new_text)
predictions = loaded_model.predict(new_text_tfidf)

print("Predictions:", predictions)
print("Model Accuracy:", accuracy)
print("Classification Report:\n", report)
```

Output :

Using streamlit.



Conclusion

Hindi sentiment analysis using text vectorization, SVM, model serialization, and rigorous evaluation is a powerful approach to gain insights from Hindi text data. By following these steps, one can develop a robust sentiment analysis model capable of classifying sentiment in Hindi text, opening doors to a wide range of applications in understanding and responding to sentiment within the Hindi-speaking community. This approach combines NLP and machine learning to provide valuable insights and actionable intelligence from textual data.