Github Link: https://github.com/GarvKathuria/MinorProject

Code :

1. **Code for Sentiment analysis**

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import LinearSVC

# Load the dataset
df = pd.read_csv('tweet_emotions.csv')

# Data preprocessing
X = df['content']
y = df['sentiment']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.6, random_state=42)

# Text vectorization using TF-IDF with custom preprocessing
tfidf_vectorizer = TfidfVectorizer(
    max_features=5000,
    lowercase=True,
    stop_words='english',  # Remove common English stopwords
    token_pattern=r'\b[^\d\W]+\b',  # Allow words with only letters
)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

# Train a sentiment analysis model (Linear Support Vector Classifier)
model = LinearSVC()
model.fit(X_train_tfidf, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test_tfidf)
# Now, prompt the user for input and predict the sentiment
user_input = input("Enter a text: ")
user_input_tfidf = tfidf_vectorizer.transform([user_input])
user_sentiment = model.predict(user_input_tfidf)
print("Predicted Sentiment:", user_sentiment[0])

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)

# Display the results
print("Accuracy:", accuracy)
print("Classification Report:\n", report)

# Create a heatmap for the confusion matrix with color-coding based on accuracy
plt.figure(figsize=(8, 6))
sns.heatmap(confusion, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
```

# Display the plot
plt.show()

Code Snippet:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import LinearSVC

# Load the dataset
df = pd.read_csv('tweet_emotions.csv')
```

Fig 1. Importing the required libraries and loading the dataset

```
# Data preprocessing
X = df['content']
y = df['sentiment']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.6, random_state=42)
```

Fig 2. Data Preprocessing and splitting the dataset

```
# Text vectorization using TF-IDF with custom preprocessing
tfidf_vectorizer = TfidfVectorizer(
    max_features=5000,
    lowercase=True,
    stop_words='english',  # Remove common English stopwords
    token_pattern=r'\b[^\d\W]+\b',  # Allow words with only letters
)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

Fig 3. Text vectorization using TF-IDF

```
# Text vectorization using TF-IDF with custom preprocessing
tfidf_vectorizer = TfidfVectorizer(
    max_features=5000,
    lowercase=True,
    stop_words='english',  # Remove common English stopwords
    token_pattern=r'\b[^\d\W]+\b',  # Allow words with only letters
)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

Fig 4. Training the sentiment analysis model

```python
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)

# Display the results
print("Accuracy:", accuracy)
print("Classification Report:\n", report)

# Create a heatmap for the confusion matrix with color-coding based on accuracy
plt.figure(figsize=(8, 6))
sns.heatmap(confusion, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")

# Display the plot
plt.show()
```

Fig 5. Implementation of Confusion Matrix

Output:

	tweet_id	sentiment	content
0	1956967341	empty	@tiffanylue i know i was listenin to bad habi...
1	1956967666	sadness	Layin n bed with a headache ughhhh...waitin o...
2	1956967696	sadness	Funeral ceremony...gloomy friday...
3	1956967789	enthusiasm	wants to hang out with friends SOON!
4	1956968416	neutral	@dannycastillo We want to trade with someone w...

	tweet_id	sentiment	content
0	1956967341	empty	@tiffanylue i know i was listenin to bad habi...
1	1956967666	sadness	Layin n bed with a headache ughhhh...waitin o...
2	1956967696	sadness	Funeral ceremony...gloomy friday...
3	1956967789	enthusiasm	wants to hang out with friends SOON!
4	1956968416	neutral	@dannycastillo We want to trade with someone w...
...	...	...	...
39995	1753918954	neutral	@JohnLloydTaylor
39996	1753919001	love	Happy Mothers Day All my love
39997	1753919005	love	Happy Mother's Day to all the mommies out ther...
39998	1753919043	happiness	@niariley WASSUP BEAUTIFUL!!! FOLLOW ME!! PEE...
39999	1753919049	love	@mopedronin bullet train from tokyo the gf ...

Fig 6. Dataset Used

```
C:\Users\This Pc\anaconda3\envs\Python\lib\site-packages\sklearn\svm\_classes.py:32: FutureWarning: The default value of `dual`
will change from `True` to `'auto'` in 1.5. Set the value of `dual` explicitly to suppress the warning.
  warnings.warn(

Enter a text: I am happy
Predicted Sentiment: love
```

Fig 7. Inputted I am happy

Predicted Sentiment: love

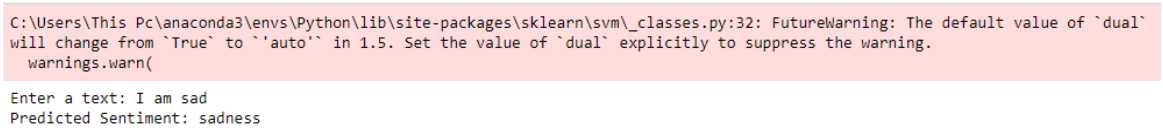

Fig 8. Inputted I am sad

Predicted Sentiment: sadness

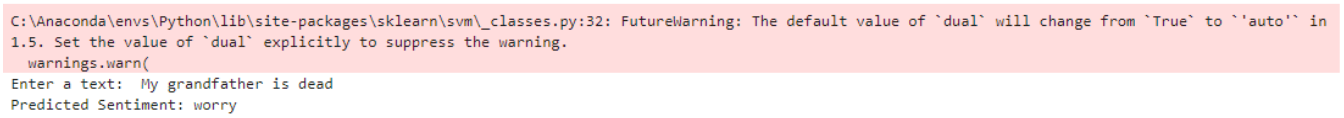

Fig 9. Inputted My grandfather is no more

Predicted Sentiment: worry

We have used the dataset(tweet_emotions.csv) in which we have over 35000+ tweets which shows some emotions. On the basis of that we will input some text asked by our code and after analyzing the dataset it will predict the sentiment.

- In the first output, we have inputted **"I am happy".** According to the dataset it representing love and it has predicted love too.
- In the second output, we have inputted **"I am sad".** According to the dataset it representing sadness and it has predicted sadness too.
- In the third output, we have inputted **"My grandfather is dead".** According to the dataset it representing sadness but it has predicted worry which also depict the sadness.
- Likewise, there are other sentiment too and it can predict that through the prompted text. For example **"I am excited"** it will predict the excitement and the happiness and it will predict that too. In our dataset there are various sentiments stored through the tweets. Above figures represents some of them.

Accuracy: 0.304125

Classification Report:

	precision	recall	f1-score	support
anger	0.00	0.00	0.00	66
boredom	0.08	0.04	0.05	103
empty	0.05	0.01	0.02	487
enthusiasm	0.05	0.01	0.02	476
fun	0.11	0.06	0.08	1032
happiness	0.29	0.31	0.30	3160
hate	0.27	0.19	0.22	784
love	0.37	0.37	0.37	2249
neutral	0.33	0.44	0.38	5174
relief	0.11	0.04	0.06	945
sadness	0.28	0.27	0.28	3106
surprise	0.10	0.05	0.07	1298
worry	0.33	0.39	0.36	5120
accuracy			0.30	24000
macro avg	0.18	0.17	0.17	24000
weighted avg	0.28	0.30	0.29	24000

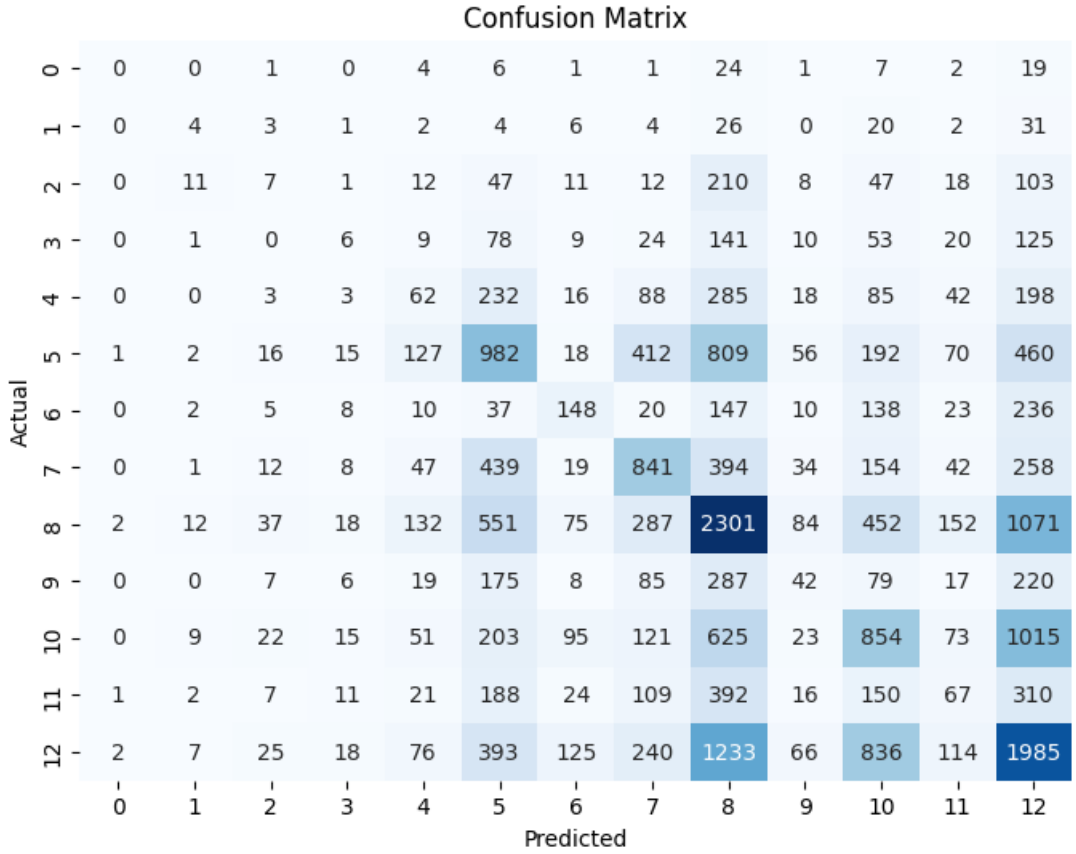

Fig 10. Confusion Matrix

The above figure is the confusion matrix of whether the model is able to predict the correct intent or not