```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, ca
```

```
dff = pd.read_csv("/content/drive/MyDrive/COLLAB/Modi_Tweets.csv" , encoding = "
dff.head()
```

	clean_text	category
0	when modi promised â□□minimum government maxim...	-1.0
1	talk all the nonsense and continue all the dra...	0.0
2	what did just say vote for modi welcome bjp t...	1.0
3	asking his supporters prefix chowkidar their n...	1.0
4	answer who among these the most powerful world...	1.0

```
import pandas as pd
import numpy as np

# Assuming your DataFrame is named df and the column is named 'category'

# Replace non-finite values with a placeholder (e.g., 0)
dff['category'].fillna(0, inplace=True)

# Convert the column to integers
dff['category'] = dff['category'].astype(int)

# If you want to specifically map float values to integer values
value_mapping = {1.0: 1, 0.0: 0, -1.0: -1}
dff['category'] = dff['category'].map(value_mapping)
```

```
dff
```

	clean_text	category
0	when modi promised â□□minimum government maxim...	-1
1	talk all the nonsense and continue all the dra...	0
2	what did just say vote for modi welcome bjp t...	1
3	asking his supporters prefix chowkidar their n...	1
4	answer who among these the most powerful world...	1
...	...	...
162975	why these 456 crores paid neerav modi not reco...	-1
162976	dear rss terrorist payal gawar what about modi...	-1
162977	did you cover her interaction forum where she ...	0
162978	there big project came into india modi dream p...	0
162979	have you ever listen about like gurukul where ...	1

162980 rows × 2 columns

```
# prompt: create a sample of random 10000 rows only

df = dff.sample(n=10000, random_state=1)
df.shape
```

```
(10000, 2)
```

XG BOOST

```
# prompt: replace the null values of clean_text

dff['clean_text'].fillna('why these 456 crores paid neerav modi ', inplace=True
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
import xgboost as xgb
from sklearn.metrics import accuracy_score

# Map the category labels to integers starting from 0
label_mapping = {-1: 0, 0: 1, 1: 2}
dff['category'] = dff['category'].map(label_mapping)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(dff['clean_text'], dff['cat
                                                    test_size=0.2, random_state

# Create TF-IDF vectors from the text data
tfidf_vectorizer = TfidfVectorizer(max_features=10000)  # Adjust max_features a
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

# Create an XGBoost classifier
xgb_classifier = xgb.XGBClassifier(objective='multi:softmax', num_class=3)  # F

# Train the classifier
xgb_classifier.fit(X_train_tfidf, y_train)

# Predict the labels for the test data
y_pred = xgb_classifier.predict(X_test_tfidf)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8671922935329488
```

```
from sklearn.model_selection import GridSearchCV

# Define the parameter grid for GridSearchCV
param_grid = {
    'learning_rate': [0.1, 0.01],
    'max_depth': [3, 5, 7],
    'n_estimators': [100, 200, 300],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}

# Initialize the XGBoost classifier
xgb_classifier = xgb.XGBClassifier(objective='multi:softmax', num_class=3)

# Perform Grid Search with cross-validation
grid_search = GridSearchCV(estimator=xgb_classifier, param_grid=param_grid, cv=
grid_search.fit(X_train_tfidf, y_train)

# Get the best estimator from Grid Search
best_xgb = grid_search.best_estimator_

# Train the best model on the full training data
best_xgb.fit(X_train_tfidf, y_train)

# Predict the labels for the test data
y_pred = best_xgb.predict(X_test_tfidf)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Best Model Accuracy:", accuracy)
```

```
# prompt: print classification report of above xgboost code

from sklearn.metrics import classification_report

# Generate the classification report
report = classification_report(y_test, y_pred)

# Print the report
print(report)
```

```
              precision    recall  f1-score   support

           0       0.91      0.69      0.79      7179
           1       0.80      0.98      0.88     11034
           2       0.92      0.86      0.89     14383

    accuracy                           0.87     32596
   macro avg       0.87      0.85      0.85     32596
weighted avg       0.88      0.87      0.86     32596
```

```
# prompt: print(y_train.dtype) convert into integer datatype

y_train = y_train.astype('int')
```

```
# prompt: generate a random forest model for dff dataset

from sklearn.ensemble import RandomForestClassifier

# Create a Random Forest classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier
rf_classifier.fit(X_train_tfidf, y_train)

# Predict the labels for the test data
y_pred = rf_classifier.predict(X_test_tfidf)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Random Forest Accuracy:", accuracy)

# Generate the classification report
report = classification_report(y_test, y_pred)

# Print the report
print(report)
```

Start coding or generate with AI.