```
import pandas as pd

# Load the CSV files
train_df = pd.read_csv('train.csv')
test_df = pd.read_csv('test.csv')

# Fill missing values
train_df['sub_category'].fillna('Unknown', inplace=True)
train_df['crimeaditionalinfo'].fillna('No additional info', inplace=True)

test_df['sub_category'].fillna('Unknown', inplace=True)
test_df['crimeaditionalinfo'].fillna('No additional info', inplace=True)

# Save the updated files
train_df.to_csv('train_filled.csv', index=False)
test_df.to_csv('test_filled.csv', index=False)

print("Missing values handled and new files saved as 'train_filled.csv' and 'test_filled.csv'")
```

```
<ipython-input-1-4767040b50bc>:8: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col


  train_df['sub_category'].fillna('Unknown', inplace=True)
<ipython-input-1-4767040b50bc>:9: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col


  train_df['crimeaditionalinfo'].fillna('No additional info', inplace=True)
<ipython-input-1-4767040b50bc>:11: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col


  test_df['sub_category'].fillna('Unknown', inplace=True)
<ipython-input-1-4767040b50bc>:12: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col


  test_df['crimeaditionalinfo'].fillna('No additional info', inplace=True)
Missing values handled and new files saved as 'train_filled.csv' and 'test_filled.csv'
```

```
train_df_filled = train_df.copy()
test_df_filled = test_df.copy()

# Identify any remaining empty cells in both files
train_empty_cells = train_df_filled.isin([""]).sum()
test_empty_cells = test_df_filled.isin([""]).sum()

train_empty_cells, test_empty_cells
```

```
(category              0
 sub_category          0
 crimeaditionalinfo    0
 dtype: int64,
 category              0
 sub_category          0
 crimeaditionalinfo    0
 dtype: int64)
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import SMOTE, RandomOverSampler
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split

# Load the train and test data
train_df = pd.read_csv('train_filled.csv')
test_df = pd.read_csv('test_filled.csv')

# Fit LabelEncoder on the union of 'sub_category' from both train and test
```

```
label_encoder = LabelEncoder()
all_labels = pd.concat([train_df['sub_category'], test_df['sub_category']]).unique()
label_encoder.fit(all_labels)

# Encode 'sub_category' in both train and test data
train_df['sub_category_encoded'] = label_encoder.transform(train_df['sub_category'])
test_df['sub_category_encoded'] = label_encoder.transform(test_df['sub_category'])

# Separate features and target for train data
X_train = train_df.drop(columns=['sub_category', 'sub_category_encoded'])
y_train = train_df['sub_category_encoded']

# One-Hot Encode 'category' column in train and test
X_train = pd.get_dummies(X_train, columns=['category'], drop_first=True)
X_test = pd.get_dummies(test_df.drop(columns=['sub_category', 'sub_category_encoded', 'crimeaditionalinfo']),
                        columns=['category'], drop_first=True)

# Align columns of X_test with X_train to ensure they have the same features
X_test = X_test.reindex(columns=X_train.columns, fill_value=0)

# TF-IDF Transformation on 'crimeaditionalinfo' for train and test
tfidf = TfidfVectorizer(max_features=500)
tfidf_train = tfidf.fit_transform(X_train['crimeaditionalinfo']).toarray()
tfidf_test = tfidf.transform(test_df['crimeaditionalinfo']).toarray()

# Convert TF-IDF result to DataFrames and concatenate with other features
tfidf_train_df = pd.DataFrame(tfidf_train, columns=[f"tfidf_{i}" for i in range(tfidf_train.shape[1])])
tfidf_test_df = pd.DataFrame(tfidf_test, columns=[f"tfidf_{i}" for i in range(tfidf_test.shape[1])])

X_train = pd.concat([X_train.reset_index(drop=True).drop(columns=['crimeaditionalinfo']), tfidf_train_df], axis=1)
X_test = pd.concat([X_test.reset_index(drop=True), tfidf_test_df], axis=1)

# Apply SMOTE with k_neighbors=1 to allow for single-sample classes, combined with RandomOverSampler
smote = SMOTE(random_state=42, k_neighbors=1)
ros = RandomOverSampler(random_state=42)

X_resampled, y_resampled = ros.fit_resample(X_train, y_train)
X_resampled, y_resampled = smote.fit_resample(X_resampled, y_resampled)

# Train a RandomForest Classifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_resampled, y_resampled)

# Make predictions on the test set
y_pred = clf.predict(X_test)

# Calculate accuracy and classification report
accuracy = accuracy_score(test_df['sub_category_encoded'], y_pred)
report = classification_report(test_df['sub_category_encoded'], y_pred, target_names=label_encoder.classes_)

print(f"Accuracy: {accuracy}")
print("Classification Report:")
print(report)


import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the train and test data
train_df = pd.read_csv('train_filled.csv')
test_df = pd.read_csv('test_filled.csv')

# Fit LabelEncoder on the union of 'sub_category' from both train and test
label_encoder = LabelEncoder()
all_labels = pd.concat([train_df['sub_category'], test_df['sub_category']]).unique()
label_encoder.fit(all_labels)

# Encode 'sub_category' in both train and test data
train_df['sub_category_encoded'] = label_encoder.transform(train_df['sub_category'])
test_df['sub_category_encoded'] = label_encoder.transform(test_df['sub_category'])

# Separate features and target for train data
X_train = train_df.drop(columns=['sub_category', 'sub_category_encoded', 'crimeaditionalinfo'])
y_train = train_df['sub_category_encoded']

# One-Hot Encode 'category' column in train and test
X_train = pd.get_dummies(X_train, columns=['category'], drop_first=True)
X_test = pd.get_dummies(test_df.drop(columns=['sub_category', 'sub_category_encoded', 'crimeaditionalinfo']),
                        columns=['category'], drop_first=True)
```

```
# Align columns of X_test with X_train to ensure they have the same features
X_test = X_test.reindex(columns=X_train.columns, fill_value=0)

# TF-IDF Transformation on 'crimeaditionalinfo' for train and test
tfidf = TfidfVectorizer(max_features=500)  # Adjust max_features if necessary
tfidf_train = tfidf.fit_transform(train_df['crimeaditionalinfo']).toarray()
tfidf_test = tfidf.transform(test_df['crimeaditionalinfo']).toarray()

# Convert TF-IDF result to DataFrames and concatenate with other features
tfidf_train_df = pd.DataFrame(tfidf_train, columns=[f"tfidf_{i}" for i in range(tfidf_train.shape[1])])
tfidf_test_df = pd.DataFrame(tfidf_test, columns=[f"tfidf_{i}" for i in range(tfidf_test.shape[1])])

X_train = pd.concat([X_train.reset_index(drop=True), tfidf_train_df], axis=1)
X_test = pd.concat([X_test.reset_index(drop=True), tfidf_test_df], axis=1)

# Train a RandomForest Classifier without oversampling
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Make predictions on the test set
y_pred = clf.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(test_df['sub_category_encoded'], y_pred)

# Generate classification report with only the classes in the test set
report = classification_report(
    test_df['sub_category_encoded'],
    y_pred,
    target_names=label_encoder.classes_[test_df['sub_category_encoded'].unique()]
)

print(f"Accuracy: {accuracy}")
print("Classification Report:")
print(report)
```

```
Accuracy: 0.7256076083127861
Classification Report:
                                                                  precision    recall  f1-score   support

                                                     Unknown       0.33      0.01      0.02        90
                          DebitCredit Card FraudSim Swap Fraud       0.48      0.45      0.46       719
                                        SQL Injection       0.00      0.00      0.00         2
                                   Fraud CallVishing       1.00      1.00      1.00       166
                                              Other       0.00      0.00      0.00         1
                              Internet Banking Related Fraud       0.61      0.83      0.70      1366
                              Unauthorised AccessData Breach       1.00      0.52      0.68        52
                                     UPI Related Frauds       1.00      0.03      0.05        39
                        Damage to computer computer systems etc       0.14      0.13      0.14       171
                                 Cheating by Impersonation       0.77      0.69      0.73      3556
                                            Malware Attack       0.40      0.01      0.02       222
                                   EWallet Related Fraud       0.13      0.14      0.14       187
                                            EMail Phishing       0.50      0.02      0.04        54
                              Profile Hacking Identity Theft       0.74      0.32      0.45      1338
                                         Data Breach/Theft       0.75      0.52      0.61       130
                                 FakeImpersonating Profile       0.55      0.47      0.51       763
                                              Email Hacking       0.61      0.31      0.41      1827
                                           Online Job Fraud       0.17      0.16      0.16       200
                                Cyber Bullying  Stalking  Sexting       0.00      0.00      0.00        13
                                         Hacking/Defacement       0.77      0.54      0.63      2973
                                       Cryptocurrency Fraud       0.00      0.00      0.00        11
                                     Online Matrimonial Fraud       0.10      0.10      0.10       170
                        Tampering with computer source documents       1.00      0.99      1.00       134
Denial of Service (DoS)/Distributed Denial of Service (DDOS) attacks       0.84      0.60      0.70       294
                                       DematDepository Fraud       0.00      0.00      0.00        38
                          Provocative Speech for unlawful acts       1.00      0.57      0.73        61
                                     Online Gambling  Betting       0.98      1.00      0.99      3670
                                            Ransomware Attack       0.63      0.58      0.60       751
                        Business Email CompromiseEmail Takeover       0.67      0.06      0.11       130
                                         Online Trafficking       0.00      0.00      0.00        18
                                            Cyber Terrorism       0.12      0.12      0.12       186
                                       Impersonating Email       0.11      0.12      0.12       167
                                 Website DefacementHacking       0.00      0.00      0.00         1
                                                 Ransomware       0.17      0.15      0.16       194
                              Computer Generated CSAM/CSEM       0.69      0.93      0.79      8890
                                         Intimidating Email       0.70      0.95      0.81       370
                                Cyber Blackmailing & Threatening       1.00      0.99      1.00      2236
                                         Sexual Harassment       0.00      0.00      0.00        39

                                                  accuracy                           0.73     31229
                                                 macro avg       0.47      0.35      0.37     31229
                                              weighted avg       0.72      0.73      0.70     31229

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined ar
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined ar
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined ar
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, classification_report
from sklearn.decomposition import PCA
from imblearn.over_sampling import RandomOverSampler
from xgboost import XGBClassifier
import numpy as np

# Load the train and test data
train_df = pd.read_csv('train_filled.csv')
test_df = pd.read_csv('test_filled.csv')

# Fit LabelEncoder on the union of 'sub_category' from both train and test
label_encoder = LabelEncoder()
all_labels = pd.concat([train_df['sub_category'], test_df['sub_category']]).unique()
label_encoder.fit(all_labels)

# Encode 'sub_category' in both train and test data
train_df['sub_category_encoded'] = label_encoder.transform(train_df['sub_category'])
test_df['sub_category_encoded'] = label_encoder.transform(test_df['sub_category'])

# Separate features and target for train data
X_train = train_df.drop(columns=['sub_category', 'sub_category_encoded', 'crimeaditionalinfo'])
y_train = train_df['sub_category_encoded']

# One-Hot Encode 'category' column in train and test
X_train = pd.get_dummies(X_train, columns=['category'], drop_first=True)
X_test = pd.get_dummies(test_df.drop(columns=['sub_category', 'sub_category_encoded', 'crimeaditionalinfo']),
                        columns=['category'], drop_first=True)

# Align columns of X_test with X_train to ensure they have the same features
X_test = X_test.reindex(columns=X_train.columns, fill_value=0)

# TF-IDF Transformation on 'crimeaditionalinfo' for train and test with reduced max_features
tfidf = TfidfVectorizer(max_features=100)
tfidf_train = tfidf.fit_transform(train_df['crimeaditionalinfo']).toarray()
tfidf_test = tfidf.transform(test_df['crimeaditionalinfo']).toarray()

# Convert TF-IDF result to DataFrames and concatenate with other features
tfidf_train_df = pd.DataFrame(tfidf_train, columns=[f"tfidf_{i}" for i in range(tfidf_train.shape[1])])
tfidf_test_df = pd.DataFrame(tfidf_test, columns=[f"tfidf_{i}" for i in range(tfidf_test.shape[1])])

X_train = pd.concat([X_train.reset_index(drop=True), tfidf_train_df], axis=1)
X_test = pd.concat([X_test.reset_index(drop=True), tfidf_test_df], axis=1)

# Apply PCA for dimensionality reduction
pca = PCA(n_components=50, random_state=42)
X_train_reduced = pca.fit_transform(X_train)
X_test_reduced = pca.transform(X_test)

# Apply Random Oversampling
ros = RandomOverSampler(random_state=42)
X_resampled, y_resampled = ros.fit_resample(X_train_reduced, y_train)

# Filter `y_resampled` and `X_resampled` to include only valid classes
valid_classes = np.unique(y_train)
indices = np.isin(y_resampled, valid_classes)
X_resampled = X_resampled[indices]
y_resampled = y_resampled[indices]

# Map `y_resampled` back to a contiguous range of classes
class_mapping = {label: idx for idx, label in enumerate(valid_classes)}
y_resampled = np.array([class_mapping[label] for label in y_resampled])

# Train a model with XGBoost for improved results
clf = XGBClassifier(objective='multi:softmax', num_class=len(valid_classes), use_label_encoder=False,
                    eval_metric='mlogloss', random_state=42)
clf.fit(X_resampled, y_resampled)

# Make predictions on the test set, using the original class encoding
y_pred = clf.predict(X_test_reduced)
y_pred = [valid_classes[label] for label in y_pred]  # Map predictions back to original labels

# Calculate accuracy and classification report
accuracy = accuracy_score(test_df['sub_category_encoded'], y_pred)
report = classification_report(test_df['sub_category_encoded'], y_pred, target_names=label_encoder.classes_)
```

```
print(f"Accuracy: {accuracy}")
print("Classification Report:")
print(report)
```

```
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [17:13:08] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-8-84a90ffc3bcc> in <cell line: 77>()
     75 # Calculate accuracy and classification report
     76 accuracy = accuracy_score(test_df['sub_category_encoded'], y_pred)
---> 77 report = classification_report(test_df['sub_category_encoded'], y_pred, target_names=label_encoder.classes_)
     78
     79 print(f"Accuracy: {accuracy}")

                              1 frames
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py in classification_report(y_true, y_pred, labels,
target_names, sample_weight, digits, output_dict, zero_division)
   2646             )
   2647         else:
-> 2648             raise ValueError(
   2649                 "Number of classes, {0}, does not match size of "
   2650                 "target_names, {1}. Try specifying the labels "

ValueError: Number of classes, 38, does not match size of target_names, 39. Try specifying the labels parameter
```

```
# Check the number of unique predicted and true classes
unique_pred_classes = np.unique(y_pred)
unique_true_classes = np.unique(test_df['sub_category_encoded'])

# Print to debug
print(f"Predicted classes: {unique_pred_classes}")
print(f"True classes: {unique_true_classes}")

# Make sure to pass only the valid classes (those actually predicted)
report = classification_report(test_df['sub_category_encoded'], y_pred,
                               labels=unique_true_classes, target_names=label_encoder.classes_[:len(unique_true_classes)])

print(f"Accuracy: {accuracy}")
print("Classification Report:")
print(report)
```

```
Predicted classes: [ 1  2  4  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 22 23 24 25 26 27
 28 29 30 31 32 34 35 36 37 38]
True classes: [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
 25 26 27 28 29 30 31 32 33 34 35 36 37 38]
Accuracy: 0.6009158154279676
Classification Report:
                                                                  precision    recall  f1-score   support

                Against Interest of sovereignty or integrity of India       0.05      0.09      0.06        90
                             Business Email CompromiseEmail Takeover       0.32      0.32      0.32       719
                                         Cheating by Impersonation       0.00      0.00      0.00         2
                                   Computer Generated CSAM/CSEM       0.98      0.95      0.96       166
                                           Cryptocurrency Fraud       0.00      0.00      0.00         1
                                   Cyber Blackmailing & Threatening       0.64      0.59      0.62      1366
                                  Cyber Bullying  Stalking  Sexting       0.14      0.04      0.06        52
                                                   Cyber Terrorism       0.24      0.18      0.21        39
                             Damage to computer computer systems etc       0.15      0.19      0.17       171
                                                 Data Breach/Theft       0.66      0.62      0.64      3556
                                DebitCredit Card FraudSim Swap Fraud       0.04      0.16      0.06       222
                                               DematDepository Fraud       0.13      0.12      0.12       187
Denial of Service (DoS)/Distributed Denial of Service (DDOS) attacks       0.20      0.09      0.13        54
                                                     EMail Phishing       0.32      0.38      0.35      1338
                                               EWallet Related Fraud       0.56      0.48      0.51       130
                                                       Email Hacking       0.34      0.34      0.34       763
                                         FakeImpersonating Profile       0.33      0.54      0.41      1827
                                                 Fraud CallVishing       0.16      0.14      0.15       200
                                                Hacking/Defacement       0.00      0.00      0.00        13
                                               Impersonating Email       0.41      0.51      0.45      2973
                                   Internet Banking Related Fraud       0.00      0.00      0.00        11
                                                Intimidating Email       0.11      0.11      0.11       170
                                                    Malware Attack       0.86      0.66      0.75       134
                                       Online Gambling  Betting       0.35      0.41      0.38       294
                                                 Online Job Fraud       0.15      0.05      0.08        38
                                       Online Matrimonial Fraud       0.29      0.03      0.06        61
                                               Online Trafficking       0.95      1.00      0.97      3670
                                                            Other       0.48      0.49      0.48       751
                                   Profile Hacking Identity Theft       0.12      0.22      0.15       130
                                Provocative Speech for unlawful acts       0.50      0.06      0.10        18
                                                        Ransomware       0.16      0.16      0.16       186
                                                 Ransomware Attack       0.10      0.14      0.12       167
                                                     SQL Injection       0.00      0.00      0.00         1
                                                Sexual Harassment       0.15      0.10      0.12       194
```

```
                 Tampering with computer source documents       0.81      0.56      0.66      8890
                                     UPI Related Frauds       0.73      0.85      0.78       370
                           Unauthorised AccessData Breach       1.00      0.99      0.99      2236
                                                  Unknown       0.27      0.10      0.15        39

                                                 accuracy                           0.60     31229
                                                macro avg       0.33      0.31      0.31     31229
                                             weighted avg       0.65      0.60      0.61     31229

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

Start coding or generate with AI.