```
# %%
import numpy as np
import pandas as pd
from collections import Counter
from sklearn.base import BaseEstimator, ClassifierMixin
from sklearn.model_selection import train_test_split
from sklearn.utils.validation import check_X_y, check_array, check_is_fitted
from sklearn.utils.multiclass import unique_labels
from sklearn.metrics import euclidean_distances, accuracy_score,
classification_report
from sklearn.feature_extraction.text import TfidfVectorizer

# %%
class KNNClassifier(BaseEstimator, ClassifierMixin):
        def __init__(self, K=3):
            self.K = K

        def fit(self, X, y):
            X, y = check_X_y(X, y)
            self.classes_ = unique_labels(y)
            self.X_ = X
            self.y_ = y
            return self

        def predict(self, X):
            check_is_fitted(self)
            X = check_array(X)

            distances = euclidean_distances(X, self.X_)
            K_nearest = np.argsort(distances, axis=1)[:, :self.K]

            K_nearest_labels = self.y_[K_nearest]
            top_labels = [Counter(row_labels).most_common(1)[0][0] for
row_labels in K_nearest_labels]
            return top_labels

# %%
base = pd.read_csv('re8.csv')
X = base['text']
y = base['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# %%
vectorizer = TfidfVectorizer()
X_train = vectorizer.fit_transform(X_train)
X_test_transformed = vectorizer.transform(X_test)

# %%
clf = KNNClassifier(K=6)

# %%
clf.fit(X_train.toarray(), y_train)
```

```
# %%
predictions = clf.predict(X_test_transformed.toarray())
accuracy = accuracy_score(y_test, predictions)
cp = classification_report(y_test, predictions)

print('Accuracy:', accuracy)
print(cp)
```