```
# %%
import gzip
import numpy as np
import pandas as pd
from sklearn.base import BaseEstimator, ClassifierMixin
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
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

# %%
class KnnGzipClassifier(BaseEstimator, ClassifierMixin):
    def __init__(self, k=5):
        self.k = k

    def ncd(self, x1, x2):
        Cx1 = len(gzip.compress(x1.encode()))
        Cx2 = len(gzip.compress(x2.encode()))
        x1x2 = " ".join([x1, x2])
        Cx1x2 = len(gzip.compress(x1x2.encode()))
        return (Cx1x2 - min(Cx1, Cx2)) / max(Cx1, Cx2)

    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)
        distance_from_x1 = []

        for x1 in X:
            for x2 in self.X_:
                ncd = self.ncd(x1, x2)
                distance_from_x1.append(ncd)

        distance_matrix = np.array(distance_from_x1).reshape(len(X), -1)
        sorted_idx = np.argsort(distance_matrix, axis=1)[:, :self.k]
        top_k_class = self.y_[sorted_idx]

        # Predict class
        predict_class = [np.argmax(np.bincount(classes)) for classes in
top_k_class]
        return np.array(predict_class)

# %%
base = pd.read_csv('Dmoz-Sports.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 = KnnGzipClassifier(k=6)
clf.fit(X_train.toarray(), y_train)

# %%
predictions = clf.predict(X_test_transformed.toarray())
accuracy = np.sum(predictions == y_test) / len(y_test)

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