Machine learning

Q-3. Imagine you have a dataset where you have different categories of data, Now you need to find the most similar data to the given data by using any 4 different similarity algorithms. Now you have to build a model which can find the most similar data to the given data.

Dataset This is the Dataset You can use this dataset for this question.

In [1]:

```
## Import the necessary libraries:-
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity, euclidean_distances, manhat

```

In [3]:

```
# Load the dataset
data = pd.read_json("Downloads/archive (3)/News_Category_Dataset_v3.json", lines=Tr
```

In [4]:

```
## Checking Top 5 Rows
data.head()
```

Out[4]:

	link	headline	category	short_description	authors
0	https://www.huffpost.com/entry/covid-boosters-...	Over 4 Million Americans Roll Up Sleeves For O...	U.S. NEWS	Health experts said it is too early to predict...	Carla K Johnson AP
1	https://www.huffpost.com/entry/american-airlin...	American Airlines Flyer Charged, Banned For Li...	U.S. NEWS	He was subdued by passengers and crew when he ...	Mary Papenfuss
2	https://www.huffpost.com/entry/funniest-tweets...	23 Of The Funniest Tweets About Cats And Dogs ...	COMEDY	"Until you have a dog you don't understand wha...	Elyse Wanshe
3	https://www.huffpost.com/entry/funniest-parent...	The Funniest Tweets From Parents This Week (Se...	PARENTING	"Accidentally put grown-up toothpaste on my to...	Caroline Bologna
4	https://www.huffpost.com/entry/amy-cooper-lose...	Woman Who Called Cops On Black Bird-Watcher Lo...	U.S. NEWS	Amy Cooper accused investment firm Franklin Te...	Nina Golgowsk

In [5]:

```
# Select relevant columns for analysis
data = data[['category', 'headline', 'short_description']]
```

In [6]:

```
# Preprocess the data
data['text'] = data['headline'] + ' ' + data['short_description']
```

In [7]:

```
# Vectorize the text data
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(data['text'])
```

In [8]:

```
# Function to find the most similar data using different similarity algorithms
def find_similar_data(query, top_n=5):
    # Vectorize the query
    query_vector = vectorizer.transform([query])

    # Calculate similarities using different algorithms
    cosine_sim = cosine_similarity(X, query_vector).flatten()
    euclidean_sim = euclidean_distances(X, query_vector).flatten()
    manhattan_sim = manhattan_distances(X, query_vector).flatten()


    # Combine similarities from different algorithms
    similarity_scores = (cosine_sim + euclidean_sim + manhattan_sim ) / 4

    # Find the indices of top similar data points
    top_indices = similarity_scores.argsort()[-top_n:][::-1]

    # Return the top similar data points
    similar_data = data.iloc[top_indices]

    return similar_data
```

In [9]:

```
# Example usage
query = "New research on climate change"
similar_data = find_similar_data(query)
print(similar_data)

```

```
         category                          headline  \
109802  WORLDPOST  Weekend Roundup: Laughing at God   
66816    POLITICS                    Sunday Roundup   
63109    POLITICS                    Sunday Roundup   
107893   POLITICS                    Sunday Roundup   
64398    POLITICS                    Sunday Roundup   

                                        short_description  \
109802  The first principle of an open society is not ...   
66816   This week the nation watched as the #NeverTrum...   
63109   This week, the nation was reminded, in ways bo...   
107893  This week began with "The Horrible Call" final...   
64398   This week started off with the horror in Orlan...   

                                                     text  
109802  Weekend Roundup: Laughing at God The first pri...  
66816   Sunday Roundup This week the nation watched as...  
63109   Sunday Roundup This week, the nation was remin...  
107893  Sunday Roundup This week began with "The Horri...  
64398   Sunday Roundup This week started off with the ...  
```

In [ ]:

```

```