Bahria University

Karachi Campus



LAB EXPERIMENT NO.

08

LIST OF TASKS

|  |  |
| --- | --- |
| **TASK NO** | **OBJECTIVE** |
| 1 | Create A music recommendation System on spotify data. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Submitted On:

**05-01-2024**

(Date: DD/MM/YY)

**Task # 01:**

**Solution:**

**CODE:**

import pandas as pd

df = pd.read\_csv("spotify\_millsongdata.csv")

A screenshot of a computer

Description automatically generateddf.head(5)

A screenshot of a computer

Description automatically generateddf.tail(5)

df.shape (57650, 4)

df.isnull().sum()

A black and white text

Description automatically generated with medium confidence

df =df.sample(5000).drop('link', axis=1).reset\_index(drop=True)

A screenshot of a computer

Description automatically generateddf.head(10)

df['text'][0]

A close-up of a text

Description automatically generated

df.shape



**Text Cleaning:**

df['text'] = df['text'].str.lower().replace(r'^\w\s', ' ').replace(r'\n', ' ', regex = True)

import nltk

from nltk.stem.porter import PorterStemmer

stemmer = PorterStemmer()

def tokenization(txt):

tokens = nltk.word\_tokenize(txt)

stemming = [stemmer.stem(w) for w in tokens]

return " ".join(stemming)

df['text'] = df['text'].apply(lambda x: tokenization(x))

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine\_similarity

tfidvector = TfidfVectorizer(analyzer='word',stop\_words='english')

matrix = tfidvector.fit\_transform(df['text'])

similarity = cosine\_similarity(matrix)

similarity[0]



df[df['song'] == 'Bring It On']



def recommendation(song\_df):

idx = df[df['song'] == song\_df].index[0]

distances = sorted(list(enumerate(similarity[idx])),reverse=True,key=lambda x:x[1])

songs = []

for m\_id in distances[1:21]:

songs.append(df.iloc[m\_id[0]].song)

return song

recommendation('Bring It On')

A screenshot of a computer screen

Description automatically generated