```
print("Name : Bhagvat Nivrutti Mutthe ")
print("Roll No : BCB-76")
print("Assignment no.2")
print("CLASSIFICATION USING DEEP NEURAL NETWORK")
```

```
Name : Bhagvat Nivrutti Mutthe
Roll No : BCB-76
Assignment no.2
CLASSIFICATION USING  DEEP  NEURAL  NETWORK
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
df=pd.read_csv("IMDB  Dataset.csv")
df1=df.head(10)
df1
```

	review	sentiment

0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive
5	Probably my all-time favorite movie, a story o...	positive
6	I sure would like to see a resurrection of a u...	positive
7	This show was an amazing, fresh & innovative i...	negative
8	Encouraged by the positive comments about this...	negative
9	If you like original gut wrenching laughter yo...	positive

Next steps: Generate code with df1 View recommended plots

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   review     50000 non-null  object
 1   sentiment  50000 non-null  object
dtypes: object(2)
memory usage: 781.4+ KB
```

```
df.isnull().sum()
```

```
review       0
sentiment    0
dtype: int64
```

```
df.sentiment.value_counts()
```

```
sentiment
positive    25000
negative    25000
Name: count, dtype: int64
```

```
df.review.value_counts().head(2)
```

```
review
Loved today's show!!! It was a variety and not solely cooking (which would have been great too). Very stimulating and
captivating, always keeping the viewer peeking around the corner to see what was coming up next. She is as down to
earth and as personable as you get, like one of us which made the show all the more enjoyable. Special guests, who are
friends as well made for a nice surprise too. Loved the 'first' theme and that the audience was invited to play along
too. I must admit I was shocked to see her come in under her time limits on a few things, but she did it and by golly
I'll be writing those recipes down. Saving time in the kitchen means more time with family. Those who haven't tuned in
yet, find out what channel and the time, I assure you that you won't be disappointed.
5
Hilarious, clean, light-hearted, and quote-worthy. What else can you ask for in a film? This is my all-time, number
one favorite movie. Ever since I was a little girl, I've dreamed of owning a blue van with flames and an observation
bubble.<br /><br />The cliché characters in ridiculous situations are what make this film such great fun. The
wonderful comedic chemistry between Stephen Furst (Harold) and Andy Tennant (Melio) make up most of my favorite parts
of the movie. And who didn't love the hopeless awkwardness of Flynch? Don't forget the airport antics of Leon's
cronies, dressed up as Hari Krishnas: dancing, chanting and playing the tambourine--unbeatable! The clues are genius,
the locations are classic, and the plot is timeless.<br /><br />A word to the wise, if you didn't watch this film when
you were little, it probably won't win a place in your heart today. But nevertheless give it a chance, you may find
that "It doesn't matter what you say, it doesn't matter what you do, you've gotta play."    4
Name: count, dtype: int64
```

```
# checking how many duplicate valu there are?
df.duplicated().value_counts()
```

```
False    49582
True       418
Name: count, dtype: int64
```

```
data=df.sample(10000)
```

```
data
```

	review	sentiment
19988	This one is a real bomb. We are supposed to be...	negative
34163	I've seen this movie today for the first time ...	negative
11383	I saw this on the Accent Underground release w...	positive
31593	Allegedly the "true story" of Juana de Castill...	negative
38089	I looked at this movie with my child eyes, and...	positive
...	...	...
47378	Sheltered young woman, home-schooled and possi...	negative
33549	When I saw the preview, I thought: this is goi...	negative
43674	This movie stinks. The stench resembles bad co...	negative
31874	Let's cut through everything in the first para...	negative
4183	The Clouded Yellow is a compact psychological ...	positive

10000 rows × 2 columns

Next steps: Generate code with data | View recommended plots

```
data.drop_duplicates(inplace=True)
```

```
data.duplicated().value_counts()
```

```
False    9986
Name: count, dtype: int64
```

```
pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.2)
```

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from bs4 import BeautifulSoup
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
# function to clean whole text
def clean_review(review, stemmer = PorterStemmer(), stop_words = set(stopwords.words("english"))):
    #removing html tags from reviews
    soup = BeautifulSoup(review, "html.parser")
    no_html_review = soup.get_text().lower()

    # empty list for adding clean words
    clean_text = []
    # cleaning stopwords and not alpha characters
    for word in review.split():
        if word not in stop_words and word.isalpha():
            clean_text.append(stemmer.stem(word))

    return " ".join(clean_text)
```

```
data.review = data.review.apply(clean_review)
```

```
#checking the clean review in specific locaion
data.review.iloc[3537]
```

```
'arthur miller alway known one great playwright work one lesser known play brought silver know great playwright arthur
miller i doubt origin play much like the movi come across empti william maci mistaken jew neighbor wwii get act i pres
um peopl behind movi probabl tri make point movi laura dern david paymer creat effect stori materi'
```

```
#how the data looks like now
data
```

	review	sentiment
19988	thi one real we suppos believ merl oberon sequ...	negative
34163	seen movi today first time i never heard proba...	negative
11383	i saw accent underground releas short i found ...	positive
31593	allegedli juana de eldest daughter cathol quee...	negative
38089	i look movi child i the stori abandon orphan b...	positive
...	...	...
47378	shelter young possibl quit harbor side come su...	negative
33549	when i saw i go great and inde could the actre...	negative
43674	thi movi the stench resembl bad cowpi sat sun ...	negative
31874	cut everyth first newest film pang brother hor...	negative
4183	the cloud yellow compact psycholog thriller in...	positive

9986 rows × 2 columns

Next steps: Generate code with data View recommended plots

```
# verctorizing reviews
#import CountVectorizer
from sklearn.feature_extraction.text import CountVectorizer

# setting max_features to 5000 to get most repeated 5000 words in reviews
cv = CountVectorizer(max_features=300,ngram_range=(1,4))
```

```
# Fitting countvectorizer in data.review and getting X for ML
X = cv.fit_transform(data.review).toarray()
x1=pd.DataFrame(X,columns=cv.get_feature_names_out())
x1
```

	absolut	act	action	actor	actual	all	almost	along	also	although	...	work	world	worst	worth	would	wri
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	1	0	
1	0	0	0	1	0	0	0	0	0	0	...	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9981	0	0	1	1	0	1	0	0	0	0	...	0	0	0	1	0	
9982	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	
9983	0	0	0	1	0	0	0	0	0	0	...	0	0	0	0	0	
9984	0	0	0	0	0	0	0	0	1	0	...	1	0	0	0	0	
9985	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	

9986 rows × 300 columns

```
x1.shape
```

```
(9986, 300)
```

```
#Importing label encoder
from sklearn.preprocessing import LabelEncoder
lb = LabelEncoder()

# positive = 1, negative = 0
data.sentiment = lb.fit_transform(data.sentiment)
```

```
data.sentiment
```

```
19988    0
34163    0
11383    1
31593    0
38089    1
        ..
47378    0
33549    0
43674    0
31874    0
4183     1
Name: sentiment, Length: 9986, dtype: int64
```

```
y=data.sentiment
```

```
y.shape
```

```
(9986,)
```

```
from sklearn.model_selection import train_test_split

# converting X, y into train test split
xtrain, xtest, ytrain, ytest = train_test_split(x1, y, test_size=0.2, random_state=42)
```

```
ytrain.shape
```

```
(7988,)
```

```
pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (24.3
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from te
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.9.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: ml-dtypes~=0.2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.25
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (24.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /us
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (
Requirement already satisfied: wrapt<1.15,>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from t
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.62.
Requirement already satisfied: tensorboard<2.16,>=2.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from ten
Requirement already satisfied: keras<2.16,>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in /usr/local/lib/python3.10/dist-packages (from tensorboa
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.1
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.1
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->t
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-o
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tens
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from werkzeug>=1.0.1->ten
Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7
```

```
pip install keras
```

```
Requirement already satisfied: keras in /usr/local/lib/python3.10/dist-packages (2.15.0)
```

```
import tensorflow.keras as tk
```

```
model = tk.Sequential()
model.add(tk.layers.Input(shape=(300,)))
model.add(tk.layers.Dense(50, activation='relu',kernel_initializer="he_uniform"))
model.add(tk.layers.Dense(1, activation='sigmoid',kernel_initializer="he_uniform"))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 50)                15050

 dense_1 (Dense)             (None, 1)                 51

=================================================================
Total params: 15101 (58.99 KB)
Trainable params: 15101 (58.99 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])
```

```
obj1=model.fit(x=xtrain,y=ytrain,epochs=80,batch_size=64,validation_data=(xtest,ytest))
```

```
Epoch 54/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0087 - accuracy: 0.9999 - val_loss: 1.0338 - val_ac
Epoch 55/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0080 - accuracy: 0.9999 - val_loss: 1.0512 - val_ac
Epoch 56/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0074 - accuracy: 0.9999 - val_loss: 1.0606 - val_ac
Epoch 57/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0069 - accuracy: 0.9999 - val_loss: 1.0809 - val_ac
Epoch 58/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0063 - accuracy: 0.9999 - val_loss: 1.1023 - val_ac
Epoch 59/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0058 - accuracy: 0.9999 - val_loss: 1.1071 - val_ac
Epoch 60/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0055 - accuracy: 0.9999 - val_loss: 1.1235 - val_ac
Epoch 61/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0050 - accuracy: 0.9999 - val_loss: 1.1413 - val_ac
Epoch 62/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0047 - accuracy: 0.9999 - val_loss: 1.1544 - val_ac
Epoch 63/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0043 - accuracy: 0.9999 - val_loss: 1.1647 - val_ac
Epoch 64/80
125/125 [==============================] - 0s 4ms/step - loss: 0.0040 - accuracy: 0.9999 - val_loss: 1.1719 - val_ac
Epoch 65/80
125/125 [==============================] - 1s 5ms/step - loss: 0.0037 - accuracy: 0.9999 - val_loss: 1.1970 - val_ac
Epoch 66/80
125/125 [==============================] - 1s 8ms/step - loss: 0.0035 - accuracy: 0.9999 - val_loss: 1.2078 - val_ac
Epoch 67/80
125/125 [==============================] - 1s 9ms/step - loss: 0.0032 - accuracy: 0.9999 - val_loss: 1.2329 - val_ac
Epoch 68/80
125/125 [==============================] - 1s 8ms/step - loss: 0.0030 - accuracy: 1.0000 - val_loss: 1.2463 - val_ac
Epoch 69/80
125/125 [==============================] - 0s 4ms/step - loss: 0.0028 - accuracy: 1.0000 - val_loss: 1.2490 - val_ac
Epoch 70/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0025 - accuracy: 1.0000 - val_loss: 1.2666 - val_ac
Epoch 71/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0024 - accuracy: 1.0000 - val_loss: 1.2757 - val_ac
Epoch 72/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0022 - accuracy: 1.0000 - val_loss: 1.2996 - val_ac
Epoch 73/80
125/125 [==============================] - 0s 3ms/step - loss: 0.0021 - accuracy: 1.0000 - val_loss: 1.3094 - val_ac
Epoch 74/80
125/125 [==============================] - 1s 7ms/step - loss: 0.0019 - accuracy: 1.0000 - val_loss: 1.3309 - val_ac
Epoch 75/80
125/125 [==============================] - 1s 5ms/step - loss: 0.0018 - accuracy: 1.0000 - val_loss: 1.3383 - val_ac
Epoch 76/80
125/125 [==============================] - 1s 7ms/step - loss: 0.0016 - accuracy: 1.0000 - val_loss: 1.3518 - val_ac
Epoch 77/80
125/125 [==============================] - 1s 6ms/step - loss: 0.0015 - accuracy: 1.0000 - val_loss: 1.3631 - val_ac
Epoch 78/80
125/125 [==============================] - 1s 9ms/step - loss: 0.0014 - accuracy: 1.0000 - val_loss: 1.3824 - val_ac
Epoch 79/80
125/125 [==============================] - 1s 9ms/step - loss: 0.0013 - accuracy: 1.0000 - val_loss: 1.3953 - val_ac
Epoch 80/80
125/125 [==============================] - 1s 9ms/step - loss: 0.0012 - accuracy: 1.0000 - val_loss: 1.4141 - val_ac
```

```
y_pred=model.predict(xtest)
```

```
63/63 [==============================] - 0s 2ms/step
```

```
y_pred
```

```
array([[0.06459261],
       [0.99981576],
       [0.94373935],
       ...,
       [0.9639384 ],
       [0.99899006],
       [1.        ]], dtype=float32)
```

```
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(ytest,y_pred.round())
```

```
accuracy
```

```
0.7577577577577578
```