In [1]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import seaborn as sns
```

In [2]:

```
from tensorflow.keras.datasets import imdb
```

```
WARNING:tensorflow:From C:\Users\jitendra\anaconda3\Anaconda\Lib\site-packa
ges\keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entro
py is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entro
py instead.
```

In [15]:

```
(x_train, y_train), (x_test,y_test) = imdb.load_data(num_words = 10000)
```

In [4]:

```
import numpy as np

def vectorize_sequences(sequences, dimensions = 10000):
  results = np.zeros((len(sequences), dimensions))
  for i,sequences in enumerate(sequences):
    results[i, sequences] = 1
  return results

x_train = vectorize_sequences(train_data)
y_train = vectorize_sequences(test_data)
```

In [5]:

```
y_train = np.asarray(train_label).astype('float32')
y_test = np.asarray(test_label).astype('float32')
```

In [6]:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

In [7]:

```
model = Sequential()
model.add(Dense(16, input_shape=(10000, ), activation = "relu"))
model.add(Dense(16, activation = "relu"))
model.add(Dense(1, activation = "sigmoid"))
```

```
WARNING:tensorflow:From C:\Users\jitendra\anaconda3\Anaconda\Lib\site-packa
ges\keras\src\backend.py:873: The name tf.get_default_graph is deprecated.
Please use tf.compat.v1.get_default_graph instead.
```

In [8]:

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

```
WARNING:tensorflow:From C:\Users\jitendra\anaconda3\Anaconda\Lib\site-packa
ges\keras\src\optimizers\__init__.py:309: The name tf.train.Optimizer is de
precated. Please use tf.compat.v1.train.Optimizer instead.
```

In [16]:

```
data = np.concatenate((x_train, x_test), axis=0)
```

```
In [17]: label = np.concatenate((y_train, y_test), axis=0)
```

```
In [19]: x_train.shape
```

```
Out[19]: (25000,)
```

```
In [20]: x_test.shape
```

```
Out[20]: (25000,)
```

```
In [21]: y_train.shape
```

```
Out[21]: (25000,)
```

```
In [22]: y_test.shape
```

```
Out[22]: (25000,)
```

```
In [24]: print("Reviews is:",x_train[0])
         print("Reviews is:",y_train[0])
```

```
Reviews is: [1, 14, 22, 16, 43, 530, 973, 1622, 1385, 65, 458, 4468, 66, 39
41, 4, 173, 36, 256, 5, 25, 100, 43, 838, 112, 50, 670, 2, 9, 35, 480, 284,
5, 150, 4, 172, 112, 167, 2, 336, 385, 39, 4, 172, 4536, 1111, 17, 546, 38,
13, 447, 4, 192, 50, 16, 6, 147, 2025, 19, 14, 22, 4, 1920, 4613, 469, 4, 2
2, 71, 87, 12, 16, 43, 530, 38, 76, 15, 13, 1247, 4, 22, 17, 515, 17, 12, 1
6, 626, 18, 2, 5, 62, 386, 12, 8, 316, 8, 106, 5, 4, 2223, 5244, 16, 480, 6
6, 3785, 33, 4, 130, 12, 16, 38, 619, 5, 25, 124, 51, 36, 135, 48, 25, 141
5, 33, 6, 22, 12, 215, 28, 77, 52, 5, 14, 407, 16, 82, 2, 8, 4, 107, 117, 5
952, 15, 256, 4, 2, 7, 3766, 5, 723, 36, 71, 43, 530, 476, 26, 400, 317, 4
6, 7, 4, 2, 1029, 13, 104, 88, 4, 381, 15, 297, 98, 32, 2071, 56, 26, 141,
6, 194, 7486, 18, 4, 226, 22, 21, 134, 476, 26, 480, 5, 144, 30, 5535, 18,
51, 36, 28, 224, 92, 25, 104, 4, 226, 65, 16, 38, 1334, 88, 12, 16, 283, 5,
16, 4472, 113, 103, 32, 15, 16, 5345, 19, 178, 32]
Reviews is: 1
```

```
In [25]: vocab=imdb.get_word_index()
```

In [26]:
```
print(vocab)
```

```
"spider'": 52016, 'hahahah': 68804, "wood'": 52017, 'transvestism': 4083
3, "hangin'": 34702, 'bringing': 2338, 'seamier': 40834, 'wooded': 34703,
'bravora': 52018, 'grueling': 16817, 'wooden': 1636, 'wednesday': 16818,
"'prix": 52019, 'altagracia': 34704, 'circuitry': 52020, 'crotch': 11585,
'busybody': 57766, "tart'n'tangy": 52021, 'burgade': 14129, 'thrace': 520
23, "tom's": 11038, 'snuggles': 52025, 'francesco': 29114, 'complainers':
52027, 'templarios': 52125, '272': 40835, '273': 52028, 'zaniacs': 52130,
'275': 34706, 'consenting': 27631, 'snuggled': 40836, 'inanimate': 15492,
'uality': 52030, 'bronte': 11926, 'errors': 4010, 'dialogs': 3230, "yomad
a's": 52031, "madman's": 34707, 'dialoge': 30585, 'usenet': 52033, 'video
drome': 40837, "kid'": 26338, 'pawed': 52034, "'girlfriend'": 30569, "'pl
easure": 52035, "'reloaded'": 52036, "kazakos'": 40839, 'rocque': 52037,
'mailings': 52038, 'brainwashed': 11927, 'mcanally': 16819, "tom''": 5203
9, 'kurupt': 25243, 'affiliated': 21905, 'babaganoosh': 52040, "noe's": 4
0840, 'quart': 40841, 'kids': 359, 'uplifting': 5034, 'controversy': 709
3, 'kida': 21906, 'kidd': 23379, "error'": 52041, 'neurologist': 52042,
'spotty': 18510, 'cobblers': 30570, 'projection': 9878, 'fastforwarding':
40842, 'sters': 52043, "eggar's": 52044, 'etherything': 52045, 'gateshea
d': 40843, 'airball': 34708, 'unsinkable': 25244, 'stern': 7180, "cerv
```

In [27]:
```
y_train
```

Out[27]:
```
array([1, 0, 0, ..., 0, 1, 0], dtype=int64)
```

In [28]:
```
y_test
```

Out[28]:
```
array([0, 1, 1, ..., 0, 0, 0], dtype=int64)
```

In [29]:
```
test_x = data[:10000]
test_y = label[:10000]
train_x = data[10000:]
train_y = label[10000:]
test_x.shape
```

Out[29]:
```
(10000,)
```

In [30]:
```
test_y.shape
```

Out[30]:
```
(10000,)
```

In [31]:
```
train_x.shape
```

Out[31]:
```
(40000,)
```

In [32]:
```
train_y.shape
```

Out[32]:
```
(40000,)
```

In [33]:
```
print("Categories:", np.unique(label))
print("Number of unique words:", len(np.unique(np.hstack(data))))
```

```
Categories: [0 1]
Number of unique words: 9998
```

In [34]:
```
length = [len(i) for i in data]
print("Average Review length:", np.mean(length))
print("Standard Deviation:", round(np.std(length)))
```

```
Average Review length: 234.75892
Standard Deviation: 173
```

In [35]:
```
print("Label:", label[0])
```

```
Label: 1
```

In [36]:
```
print("Label:", label[1])
```

```
Label: 0
```

In [37]:
```
print(data[0])
```

```
[1, 14, 22, 16, 43, 530, 973, 1622, 1385, 65, 458, 4468, 66, 3941, 4, 173,
36, 256, 5, 25, 100, 43, 838, 112, 50, 670, 2, 9, 35, 480, 284, 5, 150, 4,
172, 112, 167, 2, 336, 385, 39, 4, 172, 4536, 1111, 17, 546, 38, 13, 447,
4, 192, 50, 16, 6, 147, 2025, 19, 14, 22, 4, 1920, 4613, 469, 4, 22, 71, 8
7, 12, 16, 43, 530, 38, 76, 15, 13, 1247, 4, 22, 17, 515, 17, 12, 16, 626,
18, 2, 5, 62, 386, 12, 8, 316, 8, 106, 5, 4, 2223, 5244, 16, 480, 66, 3785,
33, 4, 130, 12, 16, 38, 619, 5, 25, 124, 51, 36, 135, 48, 25, 1415, 33, 6,
22, 12, 215, 28, 77, 52, 5, 14, 407, 16, 82, 2, 8, 4, 107, 117, 5952, 15, 2
56, 4, 2, 7, 3766, 5, 723, 36, 71, 43, 530, 476, 26, 400, 317, 46, 7, 4, 2,
1029, 13, 104, 88, 4, 381, 15, 297, 98, 32, 2071, 56, 26, 141, 6, 194, 748
6, 18, 4, 226, 22, 21, 134, 476, 26, 480, 5, 144, 30, 5535, 18, 51, 36, 28,
224, 92, 25, 104, 4, 226, 65, 16, 38, 1334, 88, 12, 16, 283, 5, 16, 4472, 1
13, 103, 32, 15, 16, 5345, 19, 178, 32]
```

In [38]:
```
index = imdb.get_word_index()
```

In [39]:
```
reverse_index = dict([(value, key) for (key, value) in index.items()])
```

In [41]:
```
decoded = " ".join( [reverse_index.get(i - 3, "#") for i in data[0]] )
print(decoded)
```

```
# this film was just brilliant casting location scenery story direction eve
ryone's really suited the part they played and you could just imagine being
there robert # is an amazing actor and now the same being director # father
came from the same scottish island as myself so i loved the fact there was
a real connection with this film the witty remarks throughout the film were
great it was just brilliant so much that i bought the film as soon as it wa
s released for # and would recommend it to everyone to watch and the fly fi
shing was amazing really cried at the end it was so sad and you know what t
hey say if you cry at a film it must have been good and this definitely was
also # to the two little boy's that played the # of norman and paul they we
re just brilliant children are often left out of the # list i think because
the stars that play them all grown up are such a big profile for the whole
film but these children are amazing and should be praised for what they hav
e done don't you think the whole story was so lovely because it was true an
d was someone's life after all that was shared with us all
```

In [44]:
```
from sklearn.model_selection import train_test_split
```

In [45]:

```
from keras.utils import to_categorical
from keras import models
from keras import layers
model = models.Sequential()
```

In [46]:

```
model.add(layers.Dense(50, activation = "relu", input_shape=(10000, )))
```

In [48]:

```
import tensorflow as tf
callback = tf.keras.callbacks.EarlyStopping(monitor='loss', patience=3)
```

In [49]:

```
model.add(layers.Dropout(0.3, noise_shape=None, seed=None))
model.add(layers.Dense(50, activation = "relu"))
model.add(layers.Dropout(0.2, noise_shape=None, seed=None))
model.add(layers.Dense(50, activation = "relu"))
model.add(layers.Dense(1, activation = "sigmoid"))
opt = tf.keras.optimizers.Adam
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_3 (Dense)             (None, 50)                500050

 dropout (Dropout)           (None, 50)                0

 dense_4 (Dense)             (None, 50)                2550

 dropout_1 (Dropout)         (None, 50)                0

 dense_5 (Dense)             (None, 50)                2550

 dense_6 (Dense)             (None, 1)                 51

 dropout_2 (Dropout)         (None, 1)                 0

 dense_7 (Dense)             (None, 50)                100
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: