Name:Nile Pallavi Sanjay(4217)

DL 2B

```
from tensorflow.keras.datasets import imdb
(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=10000)
print("Train Shape :",x_train.shape)
print("Test Shape :",x_test.shape)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb.n
17464789/17464789 [==============================] - 1s 0us/step
Train Shape : (25000,)
Test Shape : (25000,)
```

```
print("y_train shape :",y_train.shape)
print("y_test shape :",y_test.shape)
```

```
y_train shape : (25000,)
y_test shape : (25000,)
```

```
print(x_train[1])
```

```
1, 3215, 2, 4, 1153, 9, 194, 775, 7, 8255, 2, 349, 2637, 148, 605, 2, 8003, 15, 123, 125,
```

```
print(y_train[1])
```

```
0
```

```
 vocab=imdb.get_word_index()
print(vocab['the'])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb_w
1641221/1641221 [==============================] - 1s 0us/step
1
```

```
 class_names=['Negative', 'Positive']
```

```
reverse_index = dict([(value, key) for (key, value) in vocab.items()])
```

```
def decode(review):
  text=""
  for i in review:
    text=text+reverse_index[i]
    text=text+" "
  return text
decode(x_train[1])
```

```
'the thought solid thought senator do making to is spot nomination assumed while he of j
icked as getting on was did hands fact characters to always life thrillers not as me car
r of sure your way of little it strongly random to view of love it so principles of guy
er of where it of here icon film of outside to don't all unique some like of direction i
magination below keep of queen he diverse to makes this stretch and of solid it thought
tor and budget worthwhile though ok and awaiting for ever better were and diverse for bu
ed any to of making it out and follows for effects show to show cast this family us scer
ere making senator to and finds tv tend to of emerged these thing wants but and an becki
it is video do you david see scenery it in few those are of ship for with of wild to one
dark they don't do dvd with those them '
```

```
def showlen():
  print("Length of first training sample: ",len(x_train[0]))
  print("Length of second training sample: ",len(x_train[1]))
  print("Length of first test sample: ",len(x_test[0]))
  print("Length of second test sample: ",len(x_test[1]))
showlen()
```

```
Length of first training sample:  218
Length of second training sample:  189
Length of first test sample:  68
Length of second test sample:  260
```

Padding

```
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

```
x_train=pad_sequences(x_train, value=vocab['the'], padding='post', maxlen=256)
x_test=pad_sequences(x_test, value=vocab['the'], padding='post', maxlen=256)
```

```
showlen()
```

```
Length of first training sample:  256
Length of second training sample:  256
```

```
Length of first test sample:  256
Length of second test sample:  256
```

```
decode(x_train[1])
```

```
'the thought solid thought senator do making to is spot nomination assumed while he of j
icked as getting on was did hands fact characters to always life thrillers not as me car
r of sure your way of little it strongly random to view of love it so principles of guy
er of where it of here icon film of outside to don't all unique some like of direction i
magination below keep of queen he diverse to makes this stretch and of solid it thought
tor and budget worthwhile though ok and awaiting for ever better were and diverse for bu
ed any to of making it out and follows for effects show to show cast this family us scer
ere making senator to and finds tv tend to of emerged these thing wants but and an becki
it is video do you david see scenery it in few those are of ship for with of wild to one
dark they don't do dvd with those them the the the the the the the the the the the ...'
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding, GlobalAveragePooling1D
```

```
 model=Sequential()
model.add(Embedding(10000,16))
model.add(GlobalAveragePooling1D())
model.add(Dense(16,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy',metrics=['accuracy'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 16)          160000

 global_average_pooling1d (G  (None, 16)               0
 lobalAveragePooling1D)

 dense (Dense)               (None, 16)                272

 dense_1 (Dense)             (None, 1)                 17

=================================================================
Total params: 160,289
Trainable params: 160,289
Non-trainable params: 0
_________________________________________________________________
```

```
 model.fit(x_train, y_train, epochs=4, batch_size=128, verbose=1,validation_data=(x
```

```
Epoch 1/4
196/196 [==============================] - 4s 13ms/step - loss: 0.6733 - accuracy: 0.643
Epoch 2/4
196/196 [==============================] - 2s 12ms/step - loss: 0.5006 - accuracy: 0.831
Epoch 3/4
196/196 [==============================] - 2s 12ms/step - loss: 0.3406 - accuracy: 0.877
Epoch 4/4
196/196 [==============================] - 3s 17ms/step - loss: 0.2738 - accuracy: 0.899
<keras.callbacks.History at 0x7f62d7f2ebc0>
```

```
x_test[10]
```

```
array([   1, 1581,   34, 7908, 5082,   23,    6, 1374, 1120,    7,  107,
        349,    2, 1496,   11, 5116,   18,  397, 3767,    7,    4,  107,
         84, 6763,   56,   68,  456, 1402,    2,   39,    4, 1374,    9,
         35,  204,    5,   55, 4412,  212,  193,   23,    4,  326,   45,
          6, 1109,    8, 1738,    2,   15,   29,  199, 1040,    5, 2684,
         11,   14, 1403,  212, 1528,   10,   10, 2160,    2,    9,    4,
        452,   37,    2,    4,  598,  425,    5,   45, 4394,  138,   59,
        214,  467,    4, 2391,    7, 1738,    2,   19,   41, 2455, 3028,
          5, 6866, 1489,   90,  180,   18,  101, 1403,    2, 1514, 5257,
          9,    4,  564,  871,  322,   47, 2586,   27,  274,  326,    5,
          9,  150,  112,    2,   17,    6,   87,  162, 2133,   60, 3256,
         23,    4, 7999,  123,    8,   11,    2,   29,  144,   30, 2961,
       1346,    2,  214,    4,  326,    7,    2, 1496,    8, 3767,  533,
          7,  134,    2, 6229,   10,   10,    7,  265,  285,    5,  233,
         70,  593,   54,  564, 4124,    2, 1625,   27, 1546,    2,   19,
          2, 1008,   18,   89,    4,  114, 3209,    5,   45, 1139,   32,
          4,   96,  143, 3760,  958,    7,  919,    5, 7611,  367,    4,
         96,   17,   73,   17,    6,   52,  855,    7,  836,   10,   10,
         18,    2,    7,  328,  212,   14,   31,    9, 5523,    8,  591,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1], dtype=int32)
```

```
y_test[10]
```

```
1
```

```
 import numpy as np
predicted_value=model.predict(np.expand_dims(x_test[10], 0))
print(predicted_value)
if predicted_value>0.5:
  final_value=1
else:
  final_value=0
```

```
print(final_value)
print(class_names[final_value])
```

```
1/1 [==============================] - 0s 95ms/step
[[0.7985628]]
1
Positive
```

```
loss, accuracy = model.evaluate(x_test, y_test)
print("Loss :",loss)
print("Accuracy (Test Data) :",accuracy*100)
```

```
782/782 [==============================] - 2s 2ms/step - loss: 0.3008 - accuracy: 0.877
Loss : 0.30084872245788574
Accuracy (Test Data) : 87.78799772262573
```

✓ 1s completed at 12:05 PM