jupyter ImdbClassification (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [29]:
```python
import pandas as pd
import numpy as np
import re

import string
from string import digits

import numpy as np

import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical


from nltk.tokenize import word_tokenize
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

import matplotlib.pyplot as plt

lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))
```

In [30]:
```python
df = pd.read_csv("./IMDB_Dataset.csv")
df.head()
```

Out[30]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. <br /><br />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

In [31]:
```python
from sklearn import preprocessing
le =  preprocessing.LabelEncoder()
df["sentiment"] = le.fit_transform(df['sentiment'])
```

In [32]:
```python
df.head
```

Out[32]:
```
<bound method NDFrame.head of                                                   review  sentiment
0      One of the other reviewers has mentioned that ...          1
1      A wonderful little production. <br /><br />The...          1
2      I thought this was a wonderful way to spend ti...          1
3      Basically there's a family where a little boy ...          0
4      Petter Mattei's "Love in the Time of Money" is...          1
...                                                  ...        ...
49995  I thought this movie did a down right good job...          1
49996  Bad plot, bad dialogue, bad acting, idiotic di...          0
49997  I am a Catholic taught in parochial elementary...          0
49998  I'm going to have to disagree with the previou...          0
49999  No one expects the Star Trek movies to be high...          0

[50000 rows x 2 columns]>
```

In [33]:
```python
df.isnull().sum()
```

Out[33]:
```
review       0
sentiment    0
dtype: int64
```

In [34]:
```python
X = df["review"]
y = df["sentiment"]
```

In [35]:
```python
def stringprocess(text):
    text = re.sub(r"what's", "what is ", text)
    text = re.sub(r"\'s", " is", text)
    text = re.sub(r"\'ve", " have ", text)
    text = re.sub(r"can't", "cannot ", text)
    text = re.sub(r"n't", " not ", text)
    text = re.sub(r"i'm", "i am ", text)
    text = re.sub(r"\'re", " are ", text)
    text = re.sub(r"\'d", " would ", text)
    text = re.sub(r"\'ll", " will ", text)
    text = re.sub(r"\'scuse", " excuse ", text)
    text = re.sub('\W', ' ', text)
    text = re.sub('\s+', ' ', text)
    text = text.strip(' ')

    return text
```

In [36]:
```python
def  textpreprocess(text):
```

```
In [36]: def  textpreprocess(text):

    text = map(lambda x: x.lower(), text)
    text = map(lambda x: re.sub(r"https?://\S+|www\.\S+", "", x), text)
    text = map(lambda x: re.sub(re.compile(r"<.*?>|&([a-z0-9]+|#[0-9]{1,6}|#x[0-9a-f]{1,6});"),"", x), text)
    text = map(lambda x: re.sub(r'[^\x00-\x7f]',r' ', x), text)

    emoji_pattern = re.compile(
            '['
            u'\U0001F600-\U0001F64F'
            u'\U0001F300-\U0001F5FF'
            u'\U0001F680-\U0001F6FF'
            u'\U0001F1E0-\U0001F1FF'
            u'\U00002702-\U000027B0'
            u'\U000024C2-\U0001F251'
            ']+',
            flags=re.UNICODE)

    text = map(lambda x: emoji_pattern.sub(r'', x), text)
    text = map(lambda x: x.translate(str.maketrans('', '', string.punctuation)), text) # Remove punctuations


    remove_digits = str.maketrans('', '', digits)
    text = [i.translate(remove_digits) for i in text]
    text = [w for w in text if not w in stop_words]
    text = ' '.join([lemmatizer.lemmatize(w) for w in text])
    text = text.strip()
    return text
```

```
In [37]: !unzip /usr/share/nltk_data/corpora/wordnet.zip -d /usr/share/nltk_data/corpora/
```

```
'unzip' is not recognized as an internal or external command,
operable program or batch file.
```

```
In [38]: X = X.apply(lambda x: stringprocess(x))
word_tokens = X.apply(lambda x: word_tokenize(x))

preprocess_text = word_tokens.apply(lambda x: textpreprocess(x))
preprocess_text[0]
```

```
Out[38]: 'one reviewer mentioned watching  oz episode hooked right exactly happened br br first thing struck oz brutality unflinching sc
ene violence set right word go trust show faint hearted timid show pull punch regard drug sex violence hardcore classic use wor
d br br called oz nickname given oswald maximum security state penitentary focus mainly emerald city experimental section priso
n cell glass front face inwards privacy high agenda em city home many aryan muslim gangsta latino christian italian irish scuff
le death stare dodgy dealing shady agreement never far away br br would say main appeal show due fact go show would dare forget
pretty picture painted mainstream audience forget charm forget romance oz mess around first episode ever saw struck nasty surre
al could say ready watched developed taste oz got accustomed high level graphic violence violence injustice crooked guard sold
nickel inmate kill order get away well mannered middle class inmate turned prison bitch due lack street skill prison experience
watching oz may become comfortable uncomfortable viewing thats get touch darker side'
```

```
In [39]: training_portion = 0.8
train_size = int(len(preprocess_text) * training_portion)

train_data = preprocess_text[0: train_size]
train_labels = np.array(y[0: train_size])

validation_data = preprocess_text[train_size:]
validation_labels = np.array(y[train_size:])


print(len(train_data))
print(len(train_labels))
print(len(validation_data))
print(len(validation_labels))
```

```
40000
40000
10000
10000
```

```
In [40]: vocab_size = 500
oov_tok = '<OOV>'

tokenizer = Tokenizer(num_words = vocab_size, oov_token=oov_tok)
tokenizer.fit_on_texts(train_data)
word_index = tokenizer.word_index
dict(list(word_index.items())[0:10])
```

```
Out[40]: {'<OOV>': 1,
 'br': 2,
 'movie': 3,
 'film': 4,
 'one': 5,
 'like': 6,
 'would': 7,
 'time': 8,
 'good': 9,
 'character': 10}
```

```
In [41]: train_sequences = tokenizer.texts_to_sequences(train_data)
print(train_sequences[10])
```

```
[1, 1, 5, 1, 4, 1, 332, 96, 1, 172, 153, 1, 1, 2, 2, 25, 1, 94, 69, 3, 1, 59, 285, 1, 69, 1, 2, 2, 251, 217, 4, 1, 42, 183, 94,
121, 10, 1, 313, 439, 2, 2, 1, 4, 7, 1, 1, 1, 1, 2, 2, 57, 1, 51, 124, 305, 73, 1]
```

```
In [42]: embedding_dim = 50
max_length = 70
trunc_type = 'post'
```

```
padding_type = 'post'
```

In [43]:
```
train_padded = pad_sequences(train_sequences, maxlen=max_length, padding=padding_type, truncating=trunc_type)
print(len(train_sequences[0]))
print(len(train_padded[0]))
```

```
170
70
```

In [44]:
```
train_padded[0]
```

Out[44]:
```
array([  5,   1,   1,  66,   1, 174,   1, 102, 494, 486,   2,   2,  25,
        28,   1,   1,   1,   1,  18, 449, 114, 102, 244,  32,   1,  26,
         1,   1,   1,  26,   1,   1,   1,   1, 266, 449,   1, 218, 254,
       244,   2,   2, 325,   1,   1, 255,   1,   1,   1,   1,   1,   1,
         1,   1, 382,   1,   1,   1,   1,   1,   1, 223,   1,   1, 200,
         1,   1, 382, 238,  39])
```

In [45]:
```
validation_sequences = tokenizer.texts_to_sequences(validation_data)
validation_padded = pad_sequences(validation_sequences, maxlen=max_length, padding=padding_type, truncating=trunc_type)

print(len(validation_sequences))
print(validation_padded.shape)
```

```
10000
(10000, 70)
```

In [46]:
```
reverse_word_index = dict([(value, key) for (key, value) in word_index.items()])

def decode_data(text):
    return ' '.join([reverse_word_index.get(i, '?') for i in text])
print(decode_data(train_padded[10]))
print('---')
print(train_data[10])
```

```
<OOV> <OOV> one <OOV> film <OOV> based around <OOV> everything rather <OOV> <OOV> br br first <OOV> pretty funny movie <OOV> fi
nd joke <OOV> funny <OOV> br br low budget film <OOV> never problem pretty interesting character <OOV> lost interest br br <OOV
> film would <OOV> <OOV> <OOV> <OOV> br br something <OOV> better try brother another <OOV> ? ? ? ? ? ? ? ? ? ? ? ?
---
phil alien one quirky film humour based around oddness everything rather actual punchlines br br first odd pretty funny movie p
rogressed find joke oddness funny anymore br br low budget film thats never problem pretty interesting character eventually los
t interest br br imagine film would appeal stoner currently partaking br br something similar better try brother another planet
```

In [47]:
```
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(vocab_size, embedding_dim),
    tf.keras.layers.LSTM(64,activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 50)          25000

 lstm (LSTM)                 (None, 64)                29440

 dense (Dense)               (None, 32)                2080

 dense_1 (Dense)             (None, 16)                528

 dense_2 (Dense)             (None, 1)                 17

=================================================================
Total params: 57,065
Trainable params: 57,065
Non-trainable params: 0
_________________________________________________________________
```

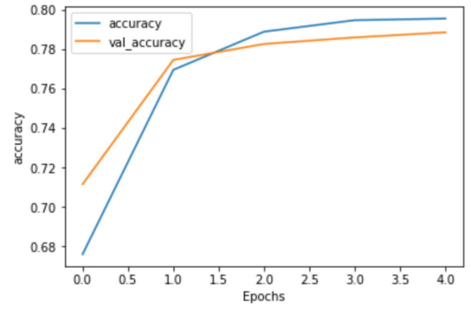
In [51]:
```
l.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

_epochs = 5
:ory = model.fit(train_padded, train_labels, epochs=num_epochs, validation_data=(validation_padded, validation_labels), verbose=2)
```

```
Epoch 1/5
1250/1250 - 152s - loss: 21.0157 - accuracy: 0.6761 - val_loss: 0.5532 - val_accuracy: 0.7116 - 152s/epoch - 121ms/step
Epoch 2/5
1250/1250 - 81s - loss: 0.4817 - accuracy: 0.7696 - val_loss: 0.4719 - val_accuracy: 0.7746 - 81s/epoch - 65ms/step
Epoch 3/5
1250/1250 - 87s - loss: 0.4479 - accuracy: 0.7889 - val_loss: 0.4567 - val_accuracy: 0.7827 - 87s/epoch - 70ms/step
Epoch 4/5
1250/1250 - 82s - loss: 0.4388 - accuracy: 0.7947 - val_loss: 0.4582 - val_accuracy: 0.7860 - 82s/epoch - 66ms/step
Epoch 5/5
1250/1250 - 83s - loss: 0.4360 - accuracy: 0.7955 - val_loss: 0.4671 - val_accuracy: 0.7886 - 83s/epoch - 66ms/step
```

In [53]:
```
def plot_graphs(history, string):
    plt.plot(history.history[string])
    plt.plot(history.history['val_'+string])
    plt.xlabel("Epochs")
    plt.ylabel(string)
    plt.legend([string, 'val_'+string])
    plt.show()
```

```
plot_graphs(history, "accuracy")
plot_graphs(history, "loss")
```


accuracy
val_accuracy
accuracy
Epochs