dl-lab-assignment-1

December 12, 2023

1. Explore Tensor Flow and Keras Libraries.

a) Create different tensors using tensorflow.

[1]:
```
import tensorflow as tf
import numpy as np
```

[2]:
```
#Tensor-multidimensional array, uniform datatype tensor- we are representing␣
↪data for your DL tasks tensors-immutable datatype This is the datatype used␣
↪in DL frameworks.
#Creating a simple tensor with a scalar value, which does not have any axes
tf1=tf.constant(4)
print(tf1)
```

```
tf.Tensor(4, shape=(), dtype=int32)
```

[3]:
```
#Tensor with only one axes-vector
tf2=tf.constant([2.1,3.1,4.1])
print(tf2)
```

```
tf.Tensor([2.1 3.1 4.1], shape=(3,), dtype=float32)
```

[4]:
```
tf_matrix=tf.constant([[1,2],[3,4],[5,6]])
print(tf_matrix)
```

```
tf.Tensor(
[[1 2]
 [3 4]
 [5 6]], shape=(3, 2), dtype=int32)
```

[5]:
```
tf_matrix3d=tf.
↪constant([[[0,1,2,3,4],[5,6,7,8,9]],[[10,11,12,13,14],[15,16,17,18,19]],[[20,21,22,23,24],[
print(tf_matrix3d)
```

```
tf.Tensor(
[[[ 0  1  2  3  4]
  [ 5  6  7  8  9]]

 [[10 11 12 13 14]
```

```
  [15 16 17 18 19]]

 [[20 21 22 23 24]
  [25 26 27 28 29]]], shape=(3, 2, 5), dtype=int32)
```

[6]:
```
h=tf.constant("Hello")
w=tf.constant("World")
print(tf.add(h,w))
```

```
tf.Tensor(b'HelloWorld', shape=(), dtype=string)
```

[7]:
```
c1=tf.constant(20)
c2=tf.constant(30)
print(tf.add(c1,c2))
```

```
tf.Tensor(50, shape=(), dtype=int32)
```

[8]:
```
m1=tf.constant([[1,2],[3,4]])
m2=tf.constant([[1,1],[1,1]])
print(tf.add(m1,m2))
```

```
tf.Tensor(
[[2 3]
 [4 5]], shape=(2, 2), dtype=int32)
```

[9]:
```
#Element wise multiplication
print(tf.multiply(m1,m2))
```

```
tf.Tensor(
[[1 2]
 [3 4]], shape=(2, 2), dtype=int32)
```

[10]:
```
#Matrix multiplication
matrix_mat=tf.matmul(m1,m2)
```

[11]:
```
print("Data type of tensor:", matrix_mat.dtype)
```

```
Data type of tensor: <dtype: 'int32'>
```

[12]:
```
print("Shape of tensor:", matrix_mat.shape)
```

```
Shape of tensor: (2, 2)
```

[13]:
```
print("Dimension of tensor:", matrix_mat.ndim)
```

```
Dimension of tensor: 2
```

b) Convert an image to tensor.

[14]:
```
from PIL import Image
```

[17]:
```
img=Image.open('Tiger.jpg')
img_tensor=tf.convert_to_tensor(img)
print(img_tensor)
```

```
<PIL.JpegImagePlugin.JpegImageFile image mode=RGB size=4192x2358 at
0x7D6769D52CE0>
tf.Tensor(
[[[ 31  59  37]
  [ 25  53  31]
  [ 20  50  26]
  …
  [112 115  72]
  [119 120  76]
  [113 114  70]]

 [[ 25  53  31]
  [ 24  52  29]
  [ 27  57  33]
  …
  [109 112  69]
  [114 115  73]
  [105 106  62]]

 [[ 16  44  21]
  [ 16  44  21]
  [ 21  49  26]
  …
  [116 119  76]
  [115 116  74]
  [101 102  60]]

 …

 [[ 95 127 104]
  [104 133 111]
  [123 151 129]
  …
  [ 92 191 100]
  [ 65 169  80]
  [ 35 141  51]]

 [[108 141 112]
  [118 148 120]
  [146 175 147]
  …
```

```
 [ 70 170  74]
 [ 56 161  68]
 [ 51 157  65]]

[[122 155 124]
 [131 162 131]
 [164 193 163]
 …
 [ 54 154  58]
 [ 49 154  59]
 [ 64 171  77]]], shape=(2358, 4192, 3), dtype=uint8)
```

c) Implement Different Activation functions on datasets.

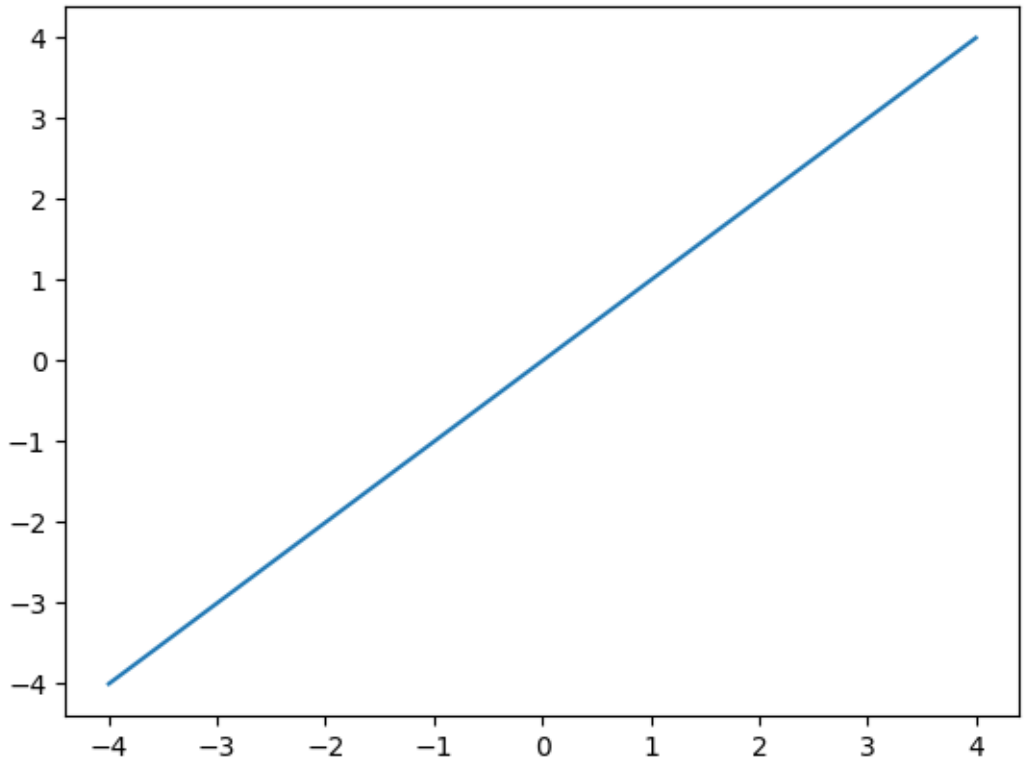
[19]:
```
import keras
import tensorflow as tf
import matplotlib.pyplot as plt
from keras import activations
```

[20]:
```
x1=tf.constant([-4.0,-2.0,0.0,2.0,4.0])
```

[21]:
```
l1=keras.activations.linear(x1)
```

[22]:
```
#plotted for
plt.plot(x1,l1)
```

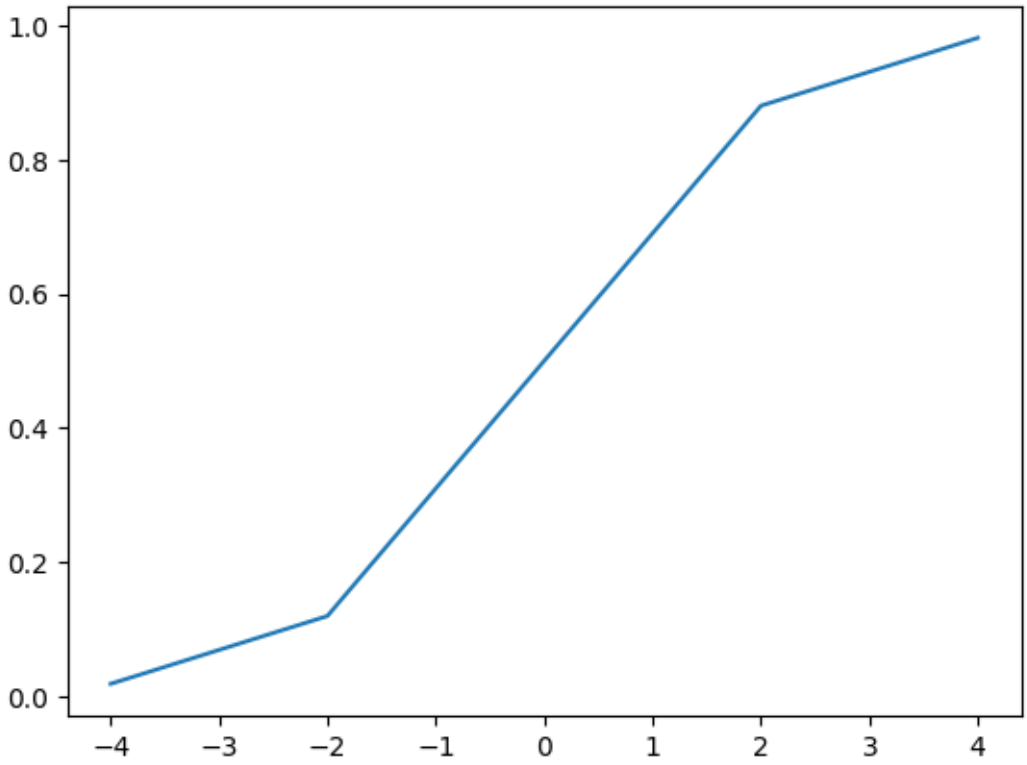
[22]:
```
[<matplotlib.lines.Line2D at 0x7d6761eaf580>]
```



[23]:
```
sigmoid_out=activations.sigmoid(x1)
```

[24]:
```
plt.plot(x1,sigmoid_out)
```

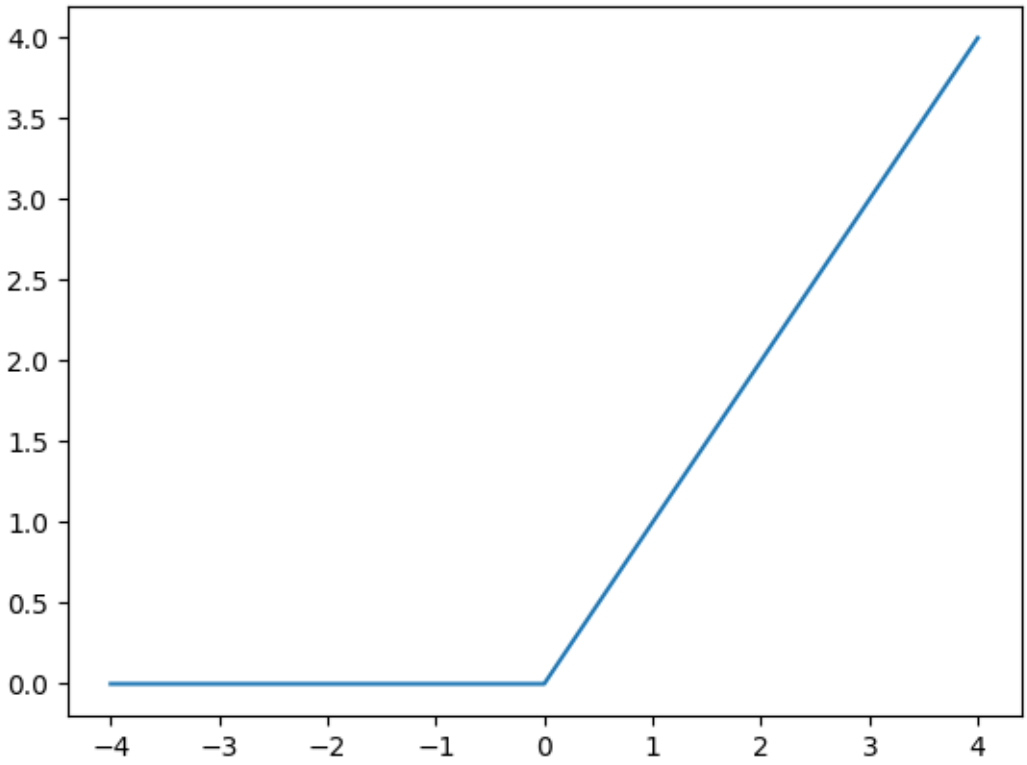
[24]:
```
[<matplotlib.lines.Line2D at 0x7d6761d77130>]
```



[25]:
```
relu_out=activations.relu(x1)
```

[26]:
```
plt.plot(x1,relu_out)
```

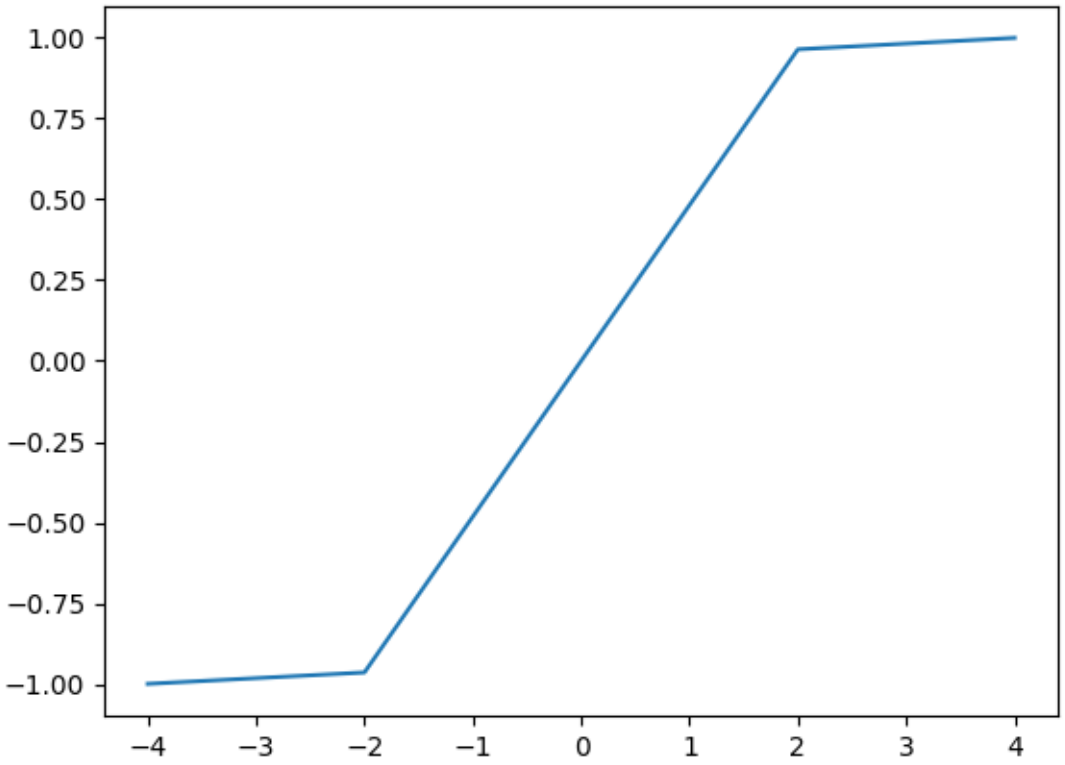
[26]:
```
[<matplotlib.lines.Line2D at 0x7d6761c42c50>]
```



[27]:
```
tanh_out=activations.tanh(x1)
```

[28]:
```
plt.plot(x1,tanh_out)
```

[28]:
```
[<matplotlib.lines.Line2D at 0x7d6761cc7d60>]
```



[29]:
```
x2=tf.constant([[4.0,5.0],[6.0,7.0]])
```

[30]:
```
softmax_out=activations.softmax(x2)
softmax_out
```

[30]:
```
<tf.Tensor: shape=(2, 2), dtype=float32, numpy=
array([[0.26894143, 0.7310586 ],
       [0.26894143, 0.7310586 ]], dtype=float32)>
```