Name : Resham Sanjay Shinalkar
Class:TE-A
Rollno:TEAD22553

Practical No:1

```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

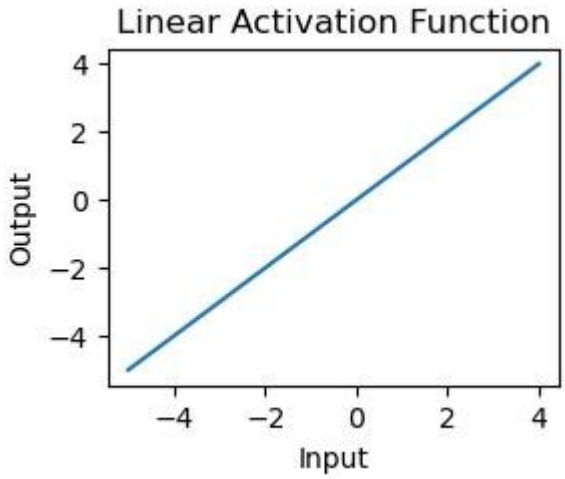
```
# Generate data for plotting x =
np.linspace(-10, 10, 400)
# Plot the functions plt.figure(figsize=(12, 8))
```

In [3]:

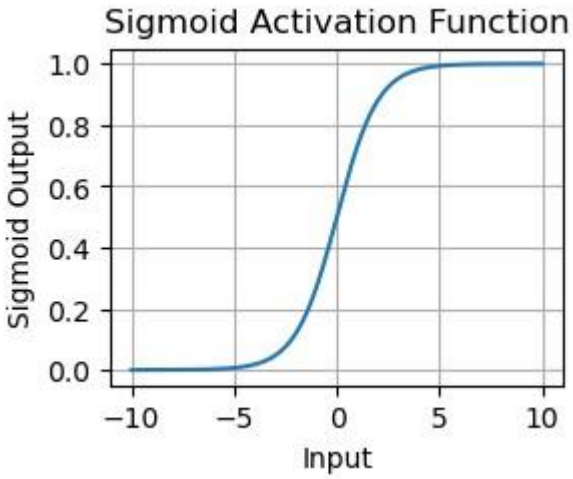
Out[3]: <Figure size 1200x800 with 0 Axes>

<Figure size 1200x800 with 0 Axes> In [4]:

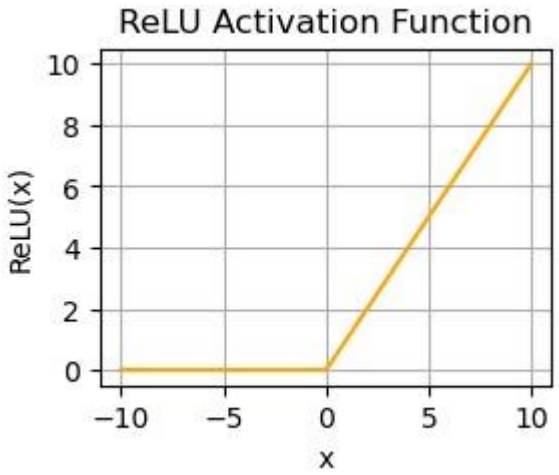
```
def linear_activation(x):         return x x_values
= range(-5, 5) y_values = [linear_activation(x) for
x in x_values] plt.subplot(2, 2, 1)
plt.plot(x_values, y_values) plt.xlabel('Input')
plt.ylabel('Output') plt.title('Linear Activation
Function') plt.show()
```



In [5]:
```
# Sigmoid function
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
plt.subplot(2, 2, 1) plt.plot(x,
sigmoid(x), label='Sigmoid')
plt.title('Sigmoid Activation Function')
plt.xlabel('Input') plt.ylabel('Sigmoid
Output') plt.grid(True)
```



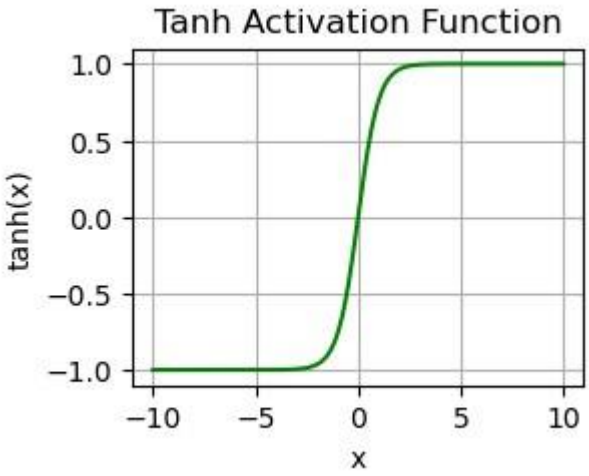
In [6]:
```
# ReLU function
def relu(x):
    return np.maximum(0, x) plt.subplot(2, 2, 2)
plt.plot(x, relu(x), label='ReLU', color='orange')
plt.title('ReLU Activation Function')
plt.xlabel("x") plt.ylabel("ReLU(x)")
plt.grid(True)
```



In [7]:

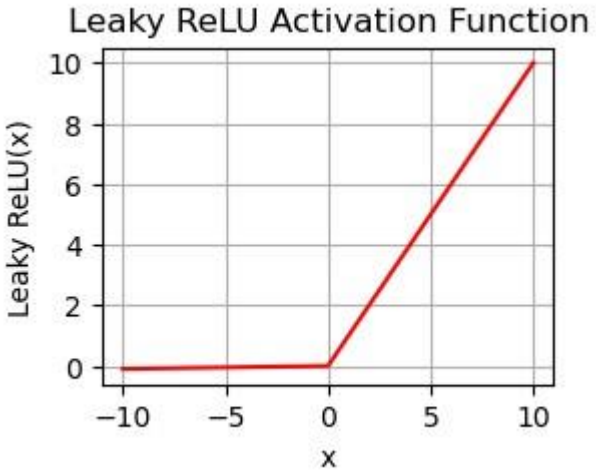
```
# Tanh function def tanh(x):   return (np.exp(x) - np.exp(-
x)) / (np.exp(x) + np.exp(-x))

plt.subplot(2, 2, 3) plt.plot(x, tanh(x),
label='Tanh', color='green') plt.title('Tanh
Activation Function') plt.xlabel("x")
plt.ylabel("tanh(x)") plt.grid(True)
```

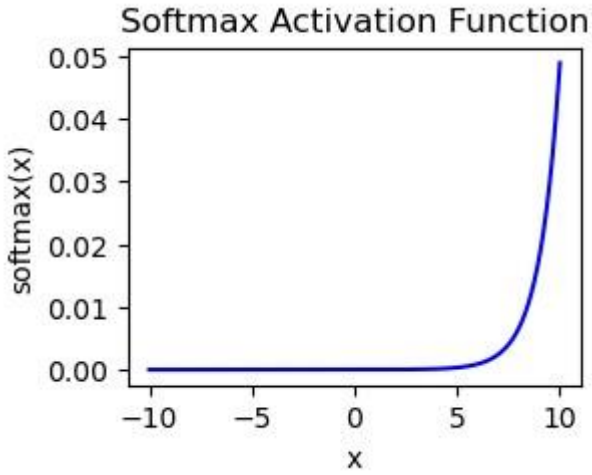


In [8]:
```
# Leaky ReLU function def leaky_relu(x,
alpha=0.01):     return np.where(x > 0,
x, alpha * x)

plt.subplot(2, 2, 4) plt.plot(x, leaky_relu(x), label='Leaky
ReLU', color='red') plt.title('Leaky ReLU Activation
Function') plt.xlabel("x") plt.ylabel("Leaky ReLU(x)")
plt.grid(True)
```

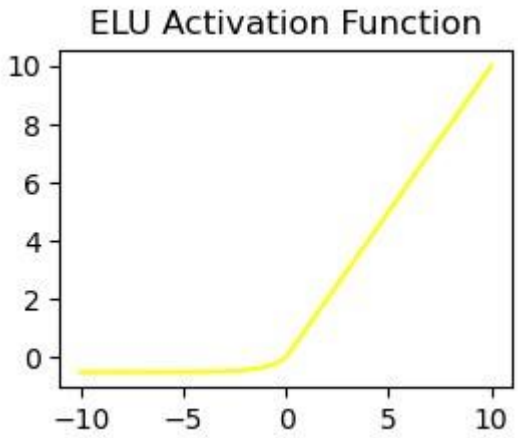


In [9]:
```
def softmax(x):
    return np.exp(x) / np.sum(np.exp(x))
plt.subplot(2, 2, 4) plt.plot(x,
softmax(x),label='softmax', color='blue')
plt.title('Softmax Activation Function')
plt.xlabel("x") plt.ylabel("softmax(x)") plt.show()
```



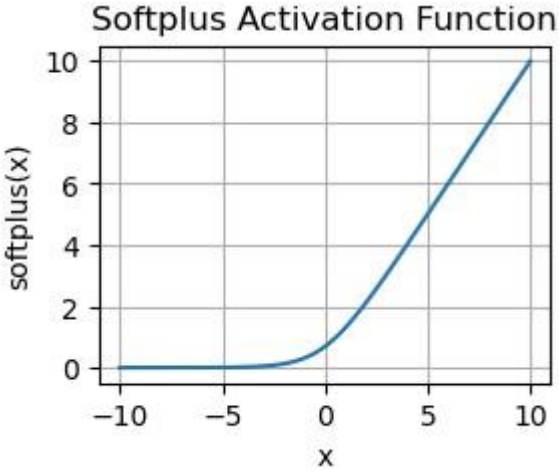
In [10]:

```
def elu(x):
lst=[]     for
i in x:
if i<0:
            lst.append(0.5*(np.exp(i) -1))
else:             lst.append(i)     return lst
plt.subplot(2, 2, 4) plt.plot(x,
elu(x),label='elu', color='yellow')
plt.title('ELU Activation Function') plt.show()
```



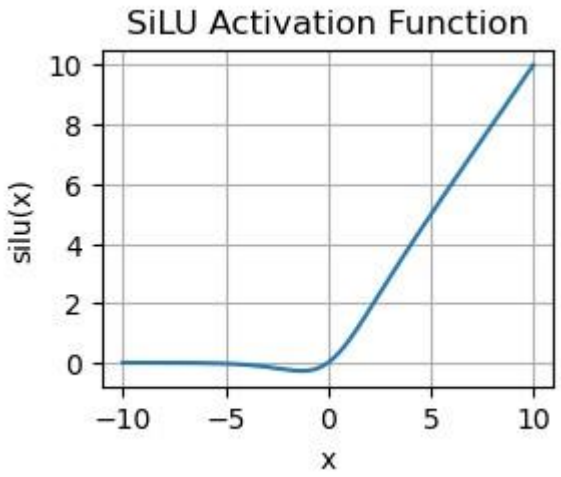
In [11]:

```
def softplus(x):
    return np.log(1 + np.exp(x))
plt.subplot(2, 2, 4) plt.plot(x,
softplus(x)) plt.xlabel('x')
plt.ylabel('softplus(x)')
plt.title('Softplus Activation Function')
plt.grid(True) plt.show()
```



In [12]:

```
#SiLU (Sigmoid Linear Unit) activation function def
silu(x):
    return x / (1 + np.exp(-x))
# Plot the graph plt.subplot(2, 2, 4)
plt.plot(x, silu(x)) plt.xlabel('x')
plt.ylabel('silu(x)') plt.title('SiLU
Activation Function') plt.grid(True)
plt.show()
```



In [ ]: