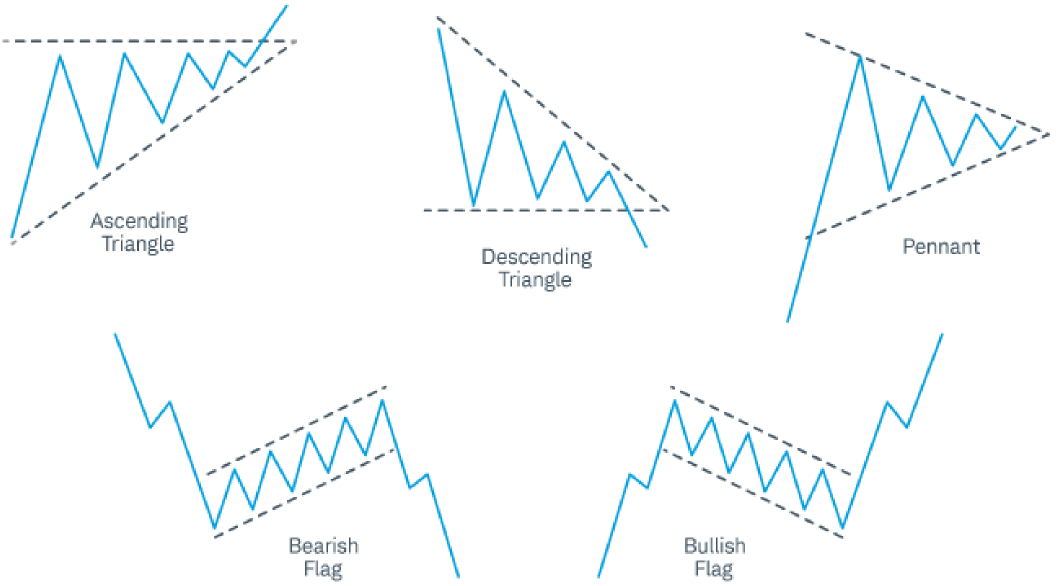
In [112…

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

Stock Chart Pattern recognition with Deep Learning



Load the Dataset

In [113…

```
stockPath = hidden
```

In [114…

```
img = (128,128)
batch = 5600
data_train = tf.keras.utils.image_dataset_from_directory(
    stockPath,
    image_size=img,
    batch_size= batch,
)
```

```
Found 1419 files belonging to 2 classes.
```

In [115…

```
data_train.class_names
```

Out[115]:

```
['ascending triangle', 'cupandHandel']
```

Tensor to Numpy

In [116…

```
import tensorflow_datasets as tfdf
```

In [117…

```
x_data = None
x_lables = None
for images,lables in tfdf.as_numpy(data_train):
  x_data = images
  x_lables = lables
```

In [118…

```
x_lables
```

Out[118]:

```
array([0, 1, 0, ..., 0, 0, 0])
```

In [119…

```
x_data.shape , x_lables.shape
```

Out[119]:

```
((1419, 128, 128, 3), (1419,))
```

In [121…

```
normalized_images = x_data / 255.0
fig, axs = plt.subplots(1, 4)

for i in range(4):
    axs[i].imshow(normalized_images[i])
    axs[i].axis('off')

plt.tight_layout()

# Show the plot
plt.show()
```

In [8]:

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

Model Declartion

In [21]:

```
def classifyImage(H=128,W=128,C=3):
  model = Sequential()
  model.add(layers.Conv2D(128,3,padding="valid",input_shape=(H,W,C)))
  model.add(layers.MaxPool2D())
  model.add(layers.Conv2D(128*2,3,padding="valid"))
  model.add(layers.MaxPool2D())
  model.add(layers.Conv2D(128*3,3,padding="valid"))
  model.add(layers.MaxPool2D())
  model.add(layers.Flatten())
  model.add(layers.Dense(128))
  model.add(layers.Dropout(0.4))
  model.add(layers.Dense(1,activation="sigmoid"))
  return model
```

In [22]:

```
classifier = classifyImage()
classifier.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_6 (Conv2D)           (None, 126, 126, 128)     3584      
                                                                 
 max_pooling2d_6 (MaxPooling  (None, 63, 63, 128)      0         
 2D)                                                             
                                                                 
 conv2d_7 (Conv2D)           (None, 61, 61, 256)       295168    
                                                                 
 max_pooling2d_7 (MaxPooling  (None, 30, 30, 256)      0         
 2D)                                                             
                                                                 
 conv2d_8 (Conv2D)           (None, 28, 28, 384)       885120    
                                                                 
 max_pooling2d_8 (MaxPooling  (None, 14, 14, 384)      0         
 2D)                                                             
                                                                 
 flatten_2 (Flatten)         (None, 75264)             0         
                                                                 
 dense_4 (Dense)             (None, 128)               9633920   
                                                                 
 dropout_2 (Dropout)         (None, 128)               0         
                                                                 
 dense_5 (Dense)             (None, 1)                 129       
                                                                 
=================================================================
Total params: 10,817,921
Trainable params: 10,817,921
Non-trainable params: 0
_________________________________________________________________
```

In [24]:
```
classifier.compile(optimizer='adam',
              loss=tf.keras.losses.BinaryCrossentropy())
```

In [15]:
```
x_data = x_data / 255.
```

In [26]:
```
epochs=20
history = classifier.fit(
  x_data,
  x_lables,
  epochs=epochs
)
```

```
Epoch 1/20
45/45 [==============================] - 87s 2s/step - loss: 2.1363
Epoch 2/20
45/45 [==============================] - 90s 2s/step - loss: 0.6501
Epoch 3/20
45/45 [==============================] - 89s 2s/step - loss: 0.6026
Epoch 4/20
45/45 [==============================] - 89s 2s/step - loss: 0.5638
Epoch 5/20
45/45 [==============================] - 90s 2s/step - loss: 0.5373
Epoch 6/20
45/45 [==============================] - 94s 2s/step - loss: 0.4938
Epoch 7/20
45/45 [==============================] - 94s 2s/step - loss: 0.4506
Epoch 8/20
45/45 [==============================] - 96s 2s/step - loss: 0.4051
Epoch 9/20
45/45 [==============================] - 99s 2s/step - loss: 0.3670
Epoch 10/20
45/45 [==============================] - 100s 2s/step - loss: 0.3174
Epoch 11/20
45/45 [==============================] - 98s 2s/step - loss: 0.2601
Epoch 12/20
45/45 [==============================] - 99s 2s/step - loss: 0.2180
Epoch 13/20
45/45 [==============================] - 99s 2s/step - loss: 0.1757
Epoch 14/20
45/45 [==============================] - 99s 2s/step - loss: 0.1504
Epoch 15/20
45/45 [==============================] - 100s 2s/step - loss: 0.1342
Epoch 16/20
45/45 [==============================] - 100s 2s/step - loss: 0.1163
Epoch 17/20
45/45 [==============================] - 100s 2s/step - loss: 0.1398
Epoch 18/20
45/45 [==============================] - 101s 2s/step - loss: 0.1124
Epoch 19/20
45/45 [==============================] - 102s 2s/step - loss: 0.1120
Epoch 20/20
45/45 [==============================] - 103s 2s/step - loss: 0.0950
```

In [30]:

```
classifier.predict(x_data)
```

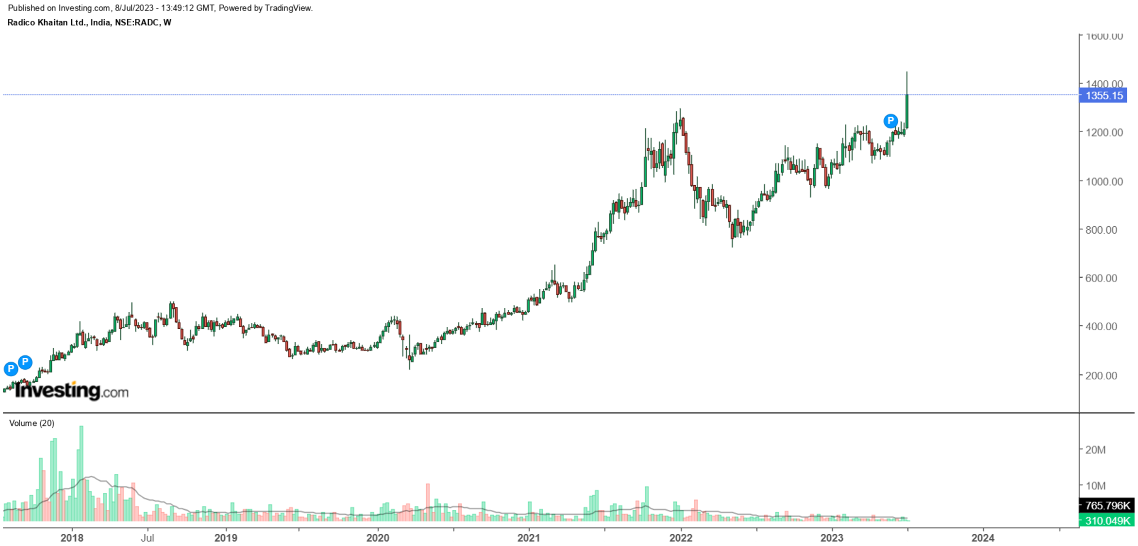
```
45/45 [==============================] - 28s 620ms/step
```

Out[30]:

```
array([[0.9994819 ],
       [0.11222942],
       [0.02221154],
       ...,
       [0.003555  ],
       [0.9986117 ],
       [0.02622554]], dtype=float32)
```

In [31]:

```
from PIL import Image
```



In [87]:
```python
image = Image.open(hide).resize((128,128))
```

In [88]:
```python
test_data = np.asarray(image)
```

In [89]:
```python
test_data.shape
```

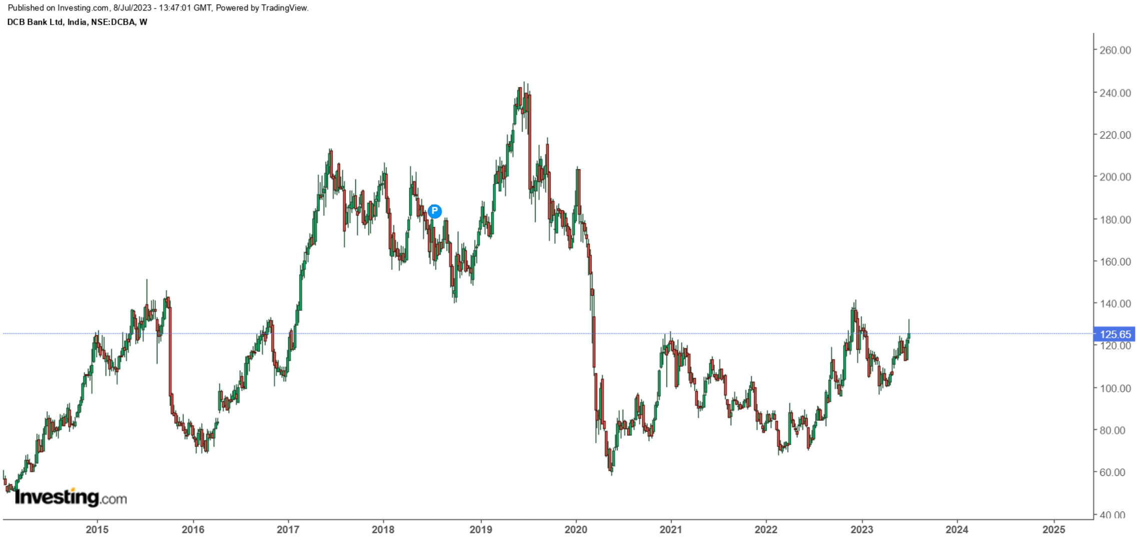
Out[89]:
```
(128, 128, 3)
```

In [93]:
```python
test_data = test_data.reshape(-1,128,128,3)
```

In [96]:
```python
test_data = test_data / 255.
```

In [97]:
```python
classifier.predict(test_data)
```

Out[97]:
```
1/1 [==============================] - 0s 51ms/step
array([[0.42067924]], dtype=float32)
```



In [108…
```python
#DCBBANK_2023-07-08_18-54-21
image1 = Image.open(hide).resize((128,128))
test_data1 = np.asarray(image1)
test_data1.shape
test_data1 = test_data1.reshape(-1,128,128,3)
test_data1 = test_data1 / 255.
```

In [109…

```
classifier.predict(test_data1)
```

```
1/1 [==============================] - 0s 140ms/step
```

Out[109]:

```
array([[0.03859152]], dtype=float32)
```

In [98]:

```
classifier.save("classifier.h5")
```