Dhamdhere Rukmini Ankush
Roll No 20 Class:-BE(IT)

In [1]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout
from tensorflow.keras.optimizers import SGD
import numpy as np
import random
import matplotlib.pyplot as plt
```

In [4]:
```
x_train = np.loadtxt('input.csv',delimiter = ',')
```

In [5]:
```
y_train = np.loadtxt('labels.csv',delimiter = ',')
```

In [6]:
```
x_test = np.loadtxt('input_test.csv',delimiter = ',')
```

In [10]:
```
y_test = np.loadtxt('labels_test.csv',delimiter = ',')
```

In [11]:
```
x_train=x_train.reshape(len(x_train), 100, 100,3)

y_train=y_train.reshape(len(y_train),1)

x_test=x_test.reshape(len(x_test), 100, 100,3)

y_test=y_test.reshape(len(y_test),1)

x_train=x_train/255.0

x_test = x_test/255.0
```

In [12]:
```
print("Shape of X_train:", x_train.shape)

print("Shape of Y_train:", y_train. shape)

print("Shape of X_test:", x_test.shape)

print("Shape of X_test: ", y_test.shape)
```

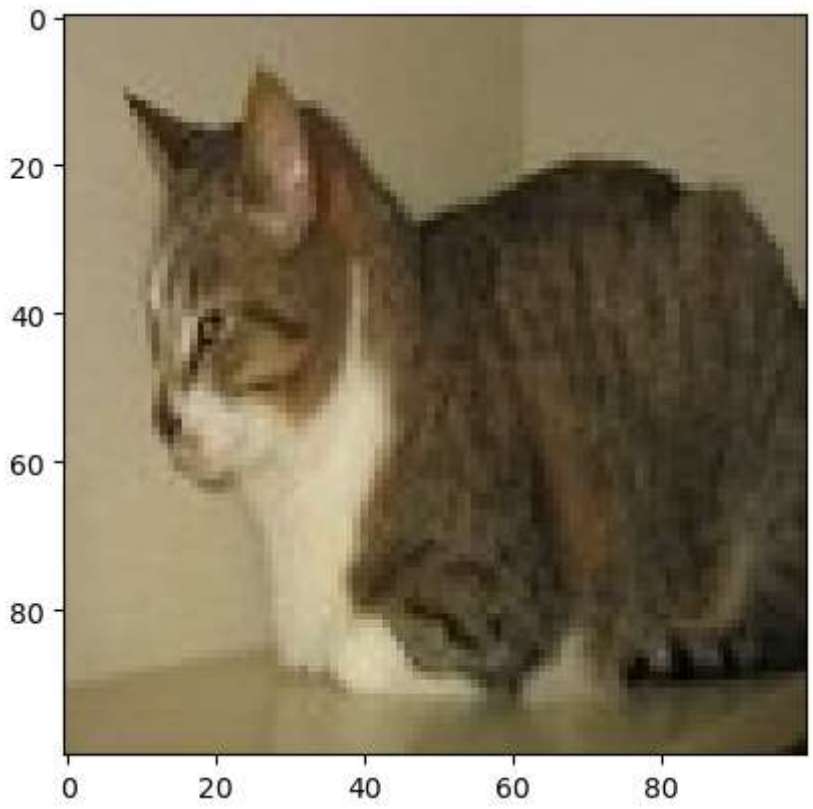
```
Shape of X_train: (2000, 100, 100, 3)
Shape of Y_train: (2000, 1)
Shape of X_test: (400, 100, 100, 3)
Shape of X_test:  (400, 1)
```

In [13]:
```
idx =random.randint(0, len(x_train))

plt.imshow(x_train[idx, :])

plt.show()
```

In [14]:
```
idx = random.randint(0, len(x_train))
plt.imshow(x_train[idx,:])
plt.show()
```



In [22]:
```
model=Sequential([
    Conv2D(256, (3,3), activation = 'relu', input_shape=(100,100,3)),
    BatchNormalization(),
    MaxPooling2D((2,2)),
    Conv2D(128, (3,3), activation = 'relu'),
    BatchNormalization(),
    MaxPooling2D((2,2)),
    Conv2D(64, (3,3), activation = 'relu'),
    BatchNormalization(),
    MaxPooling2D((2,2)),
    Flatten(),
    Dense (64, activation='relu'),
#     Dropout(0.4),
    Dense(1, activation='sigmoid')
])
```

In [25]:
```
opt=SGD(momentum=0.9)
model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

In [26]:
```
model.fit(x_train,y_train,epochs=10,batch_size=32,validation_data=(x_test,y_test)
```

```
Epoch 1/10
63/63 [==============================] - 167s 3s/step - loss: 0.7734 - accurac
y: 0.5590 - val_loss: 0.6923 - val_accuracy: 0.5400
Epoch 2/10
63/63 [==============================] - 170s 3s/step - loss: 0.5886 - accurac
y: 0.6880 - val_loss: 0.7957 - val_accuracy: 0.5100
Epoch 3/10
63/63 [==============================] - 260s 4s/step - loss: 0.5173 - accurac
y: 0.7545 - val_loss: 0.8192 - val_accuracy: 0.5350
Epoch 4/10
63/63 [==============================] - 201s 3s/step - loss: 0.4068 - accurac
y: 0.8170 - val_loss: 0.9623 - val_accuracy: 0.6100
Epoch 5/10
63/63 [==============================] - 204s 3s/step - loss: 0.3180 - accurac
y: 0.8605 - val_loss: 1.2251 - val_accuracy: 0.5800
Epoch 6/10
63/63 [==============================] - 347s 6s/step - loss: 0.2354 - accurac
y: 0.9075 - val_loss: 1.0917 - val_accuracy: 0.5850
Epoch 7/10
63/63 [==============================] - 209s 3s/step - loss: 0.1508 - accurac
y: 0.9465 - val_loss: 1.1989 - val_accuracy: 0.6425
Epoch 8/10
63/63 [==============================] - 230s 4s/step - loss: 0.1121 - accurac
y: 0.9610 - val_loss: 1.2505 - val_accuracy: 0.6500
Epoch 9/10
63/63 [==============================] - 283s 5s/step - loss: 0.0671 - accurac
y: 0.9805 - val_loss: 1.4445 - val_accuracy: 0.6600
Epoch 10/10
63/63 [==============================] - 206s 3s/step - loss: 0.0243 - accurac
y: 0.9950 - val_loss: 1.4560 - val_accuracy: 0.6550
```

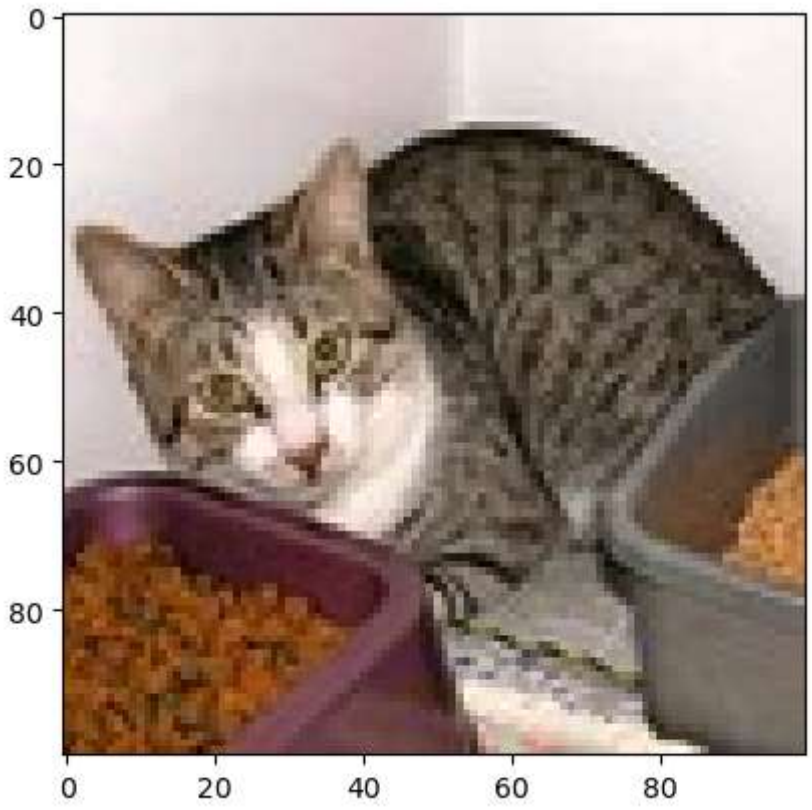
Out[26]:
```
<keras.callbacks.History at 0x22a9d324b50>
```

In [27]:
```
model.evaluate(x_test,y_test)
```

```
13/13 [==============================] - 10s 747ms/step - loss: 1.4560 - accura
cy: 0.6550
```

Out[27]:
```
[1.4559926986694336, 0.6549999713897705]
```

In [29]:
```
idx2=random.randint(0, len(y_test))
plt.imshow(x_test[idx2,:])
plt.show()
```



In [33]:
```
y_pred=model.predict(x_test[idx2,:].reshape(1,100,100,3))
y_pred = y_pred > 0.5

if(y_pred==0):
    pred ='dog'

else:
    pred ='cat'

print("Our model says it is a", pred)
```

```
1/1 [==============================] - 2s 2s/step
Our model says it is a cat
```

In [34]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 98, 98, 256)       7168

 batch_normalization_3 (Batc  (None, 98, 98, 256)      1024
 hNormalization)

 max_pooling2d_3 (MaxPooling  (None, 49, 49, 256)      0
 2D)

 conv2d_4 (Conv2D)           (None, 47, 47, 128)       295040

 batch_normalization_4 (Batc  (None, 47, 47, 128)      512
 hNormalization)

 max_pooling2d_4 (MaxPooling  (None, 23, 23, 128)      0
 2D)

 conv2d_5 (Conv2D)           (None, 21, 21, 64)        73792

 batch_normalization_5 (Batc  (None, 21, 21, 64)       256
 hNormalization)

 max_pooling2d_5 (MaxPooling  (None, 10, 10, 64)       0
 2D)

 flatten_1 (Flatten)         (None, 6400)              0

 dense_2 (Dense)             (None, 64)                409664

 dense_3 (Dense)             (None, 1)                 65

=================================================================
Total params: 787,521
Trainable params: 786,625
Non-trainable params: 896
_________________________________________________________________
```

In [35]:
```
score = model.evaluate(x_test, y_test, verbose = 0)
print("Test Score: ", score[0])
print("Test accuracy: ", score [1])
```

```
Test Score:  1.4559926986694336
Test accuracy:  0.6549999713897705
```

In [ ]:
```

```