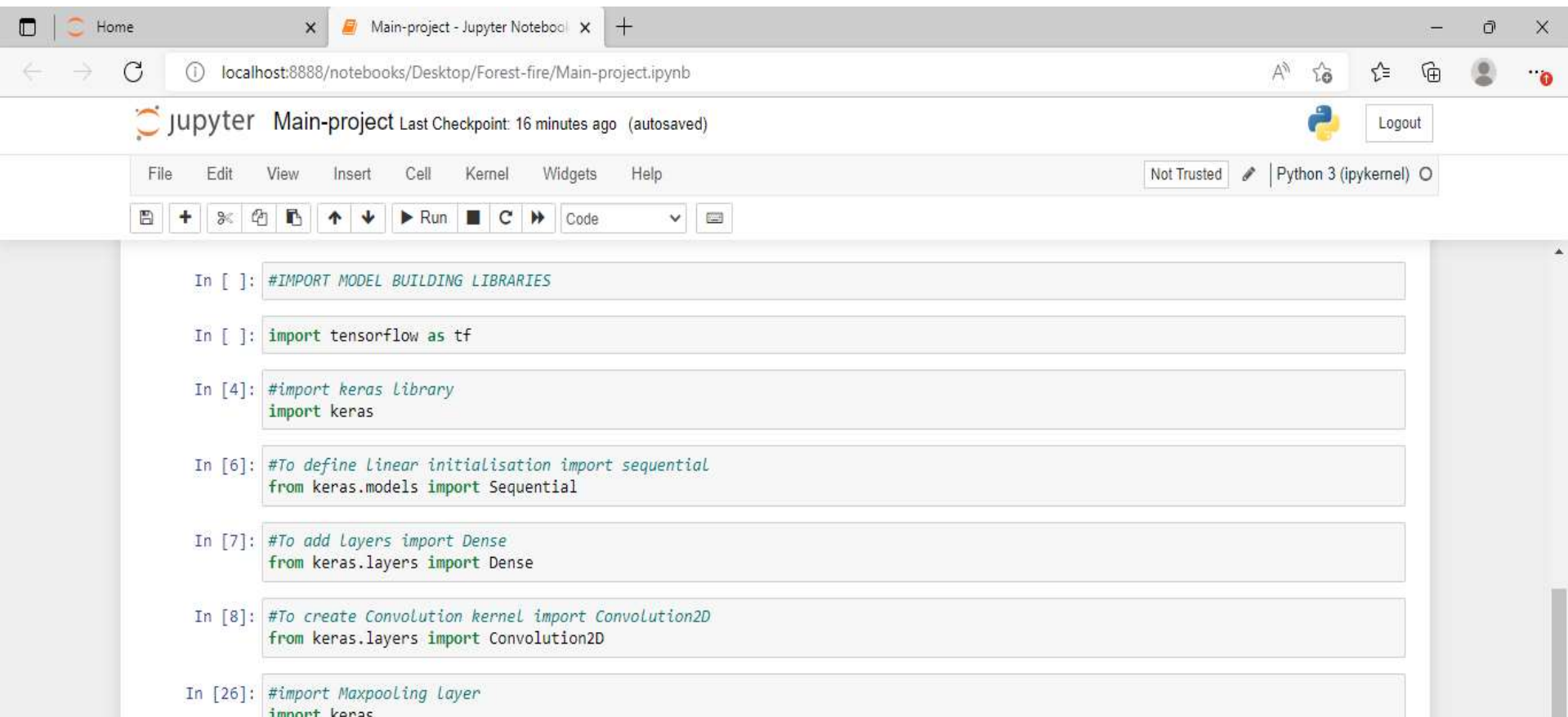


Model Building

Importing The Model Building Libraries



The screenshot shows a Jupyter Notebook interface in a web browser. The browser's address bar displays the URL `localhost:8888/notebooks/Desktop/Forest-fire/Main-project.ipynb`. The Jupyter interface includes a top bar with the 'jupyter' logo, the name of the project 'Main-project', and a 'Logout' button. Below this is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A status bar on the right indicates 'Not Trusted' and 'Python 3 (ipykernel)'. The main area contains a series of code cells, each with a prompt like 'In []:' followed by Python code for importing libraries. The code is as follows:

```
In [ ]: #IMPORT MODEL BUILDING LIBRARIES

In [ ]: import tensorflow as tf

In [4]: #import keras library
import keras

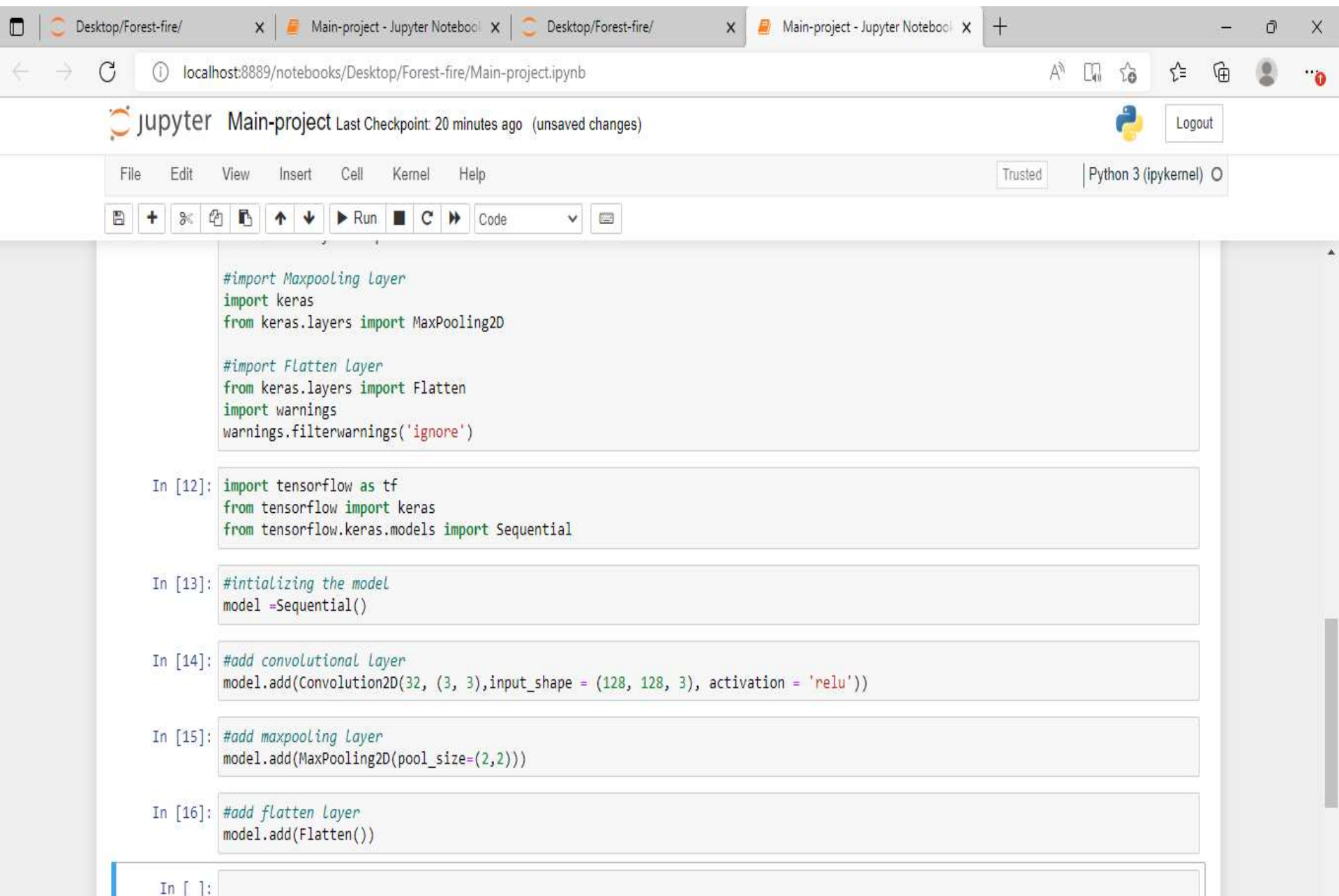
In [6]: #To define linear initialisation import sequential
from keras.models import Sequential

In [7]: #To add layers import Dense
from keras.layers import Dense

In [8]: #To create Convolution kernel import Convolution2D
from keras.layers import Convolution2D

In [26]: #import Maxpooling layer
import keras
```

Initializing The Model&Adding CNN Layers



The screenshot displays a Jupyter Notebook interface in a web browser. The browser tabs show 'Desktop/Forest-fire/' and 'Main-project - Jupyter Notebook'. The address bar indicates the URL 'localhost:8889/notebooks/Desktop/Forest-fire/Main-project.ipynb'. The Jupyter interface includes a top bar with the 'jupyter' logo, the name 'Main-project', and a status message 'Last Checkpoint: 20 minutes ago (unsaved changes)'. A 'Logout' button is visible. Below the top bar is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help'. To the right of the menu bar are 'Trusted' and 'Python 3 (ipykernel)' indicators. A toolbar with various icons for file operations and execution is located below the menu bar. The main area contains several code cells. The first cell is a code block with comments and imports for MaxPooling2D, Flatten, and warnings. Subsequent cells are labeled 'In [12]:', 'In [13]:', 'In [14]:', 'In [15]:', and 'In [16]:', each containing code to import tensorflow and keras, initialize a Sequential model, and add Convolution2D, MaxPooling2D, and Flatten layers respectively.

```
#import Maxpooling layer
import keras
from keras.layers import MaxPooling2D

#import Flatten Layer
from keras.layers import Flatten
import warnings
warnings.filterwarnings('ignore')
```

```
In [12]: import tensorflow as tf
        from tensorflow import keras
        from tensorflow.keras.models import Sequential
```

```
In [13]: #initializing the model
        model =Sequential()
```

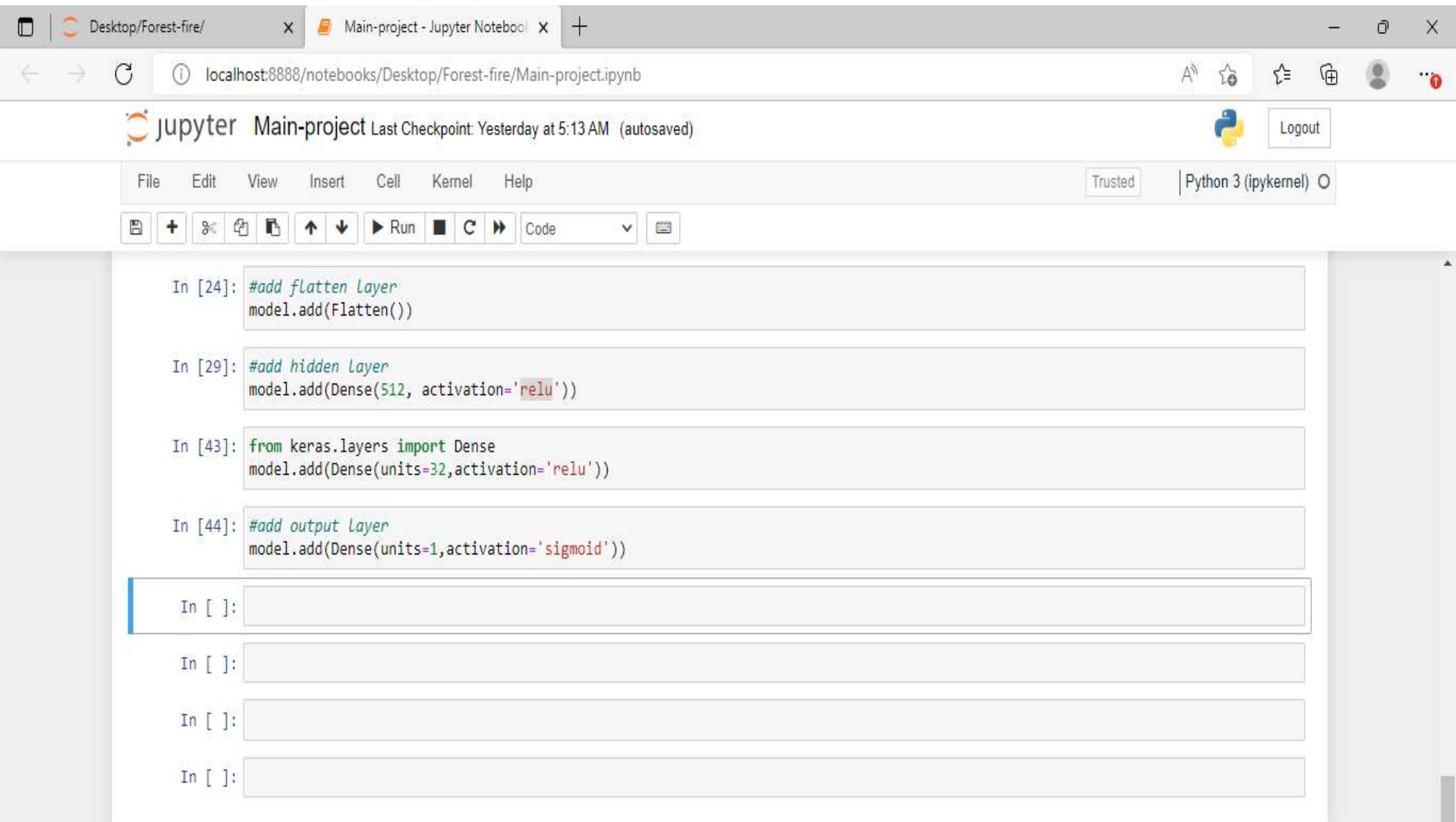
```
In [14]: #add convolutional layer
        model.add(Convolution2D(32, (3, 3),input_shape = (128, 128, 3), activation = 'relu'))
```

```
In [15]: #add maxpooling layer
        model.add(MaxPooling2D(pool_size=(2,2)))
```

```
In [16]: #add flatten layer
        model.add(Flatten())
```

```
In [ ]:
```

Adding Dense Layers



The screenshot displays a Jupyter Notebook interface within a web browser. The browser's address bar shows the URL `localhost:8888/notebooks/Desktop/Forest-fire/Main-project.ipynb`. The Jupyter interface includes a top bar with the project name "Main-project", a "Logout" button, and a "Trusted" status indicator. Below this is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, and Help. A toolbar contains icons for saving, creating new cells, undo, redo, and running code. The notebook content area shows four code cells, each with a prompt like "In [24]:" followed by Python code that adds layers to a Keras model. The code uses `Flatten()` and `Dense()` layers with various units and activation functions. The fifth cell is currently selected and empty.

```
In [24]: #add flatten layer
model.add(Flatten())

In [29]: #add hidden layer
model.add(Dense(512, activation='relu'))

In [43]: from keras.layers import Dense
model.add(Dense(units=32,activation='relu'))

In [44]: #add output Layer
model.add(Dense(units=1,activation='sigmoid'))

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```

Configuring The Learning Process

```
In [65]: #configure the learning process
model.compile(loss = 'binary_crossentropy',
              optimizer = "adam",
              metrics = ["accuracy"])
```

Training The Model

In [21]: *#Training the model*

```
model.fit_generator(x_train, steps_per_epoch=14,  
                   epochs=2, validation_data=x_test,  
                   validation_steps=4)
```

Epoch 1/2

14/14 [=====] - 71s 4s/step - loss: 3.5351 - accuracy: 0.6606 - val_loss: 0.6744 - val_accuracy: 0.8430

Epoch 2/2

14/14 [=====] - 31s 2s/step - loss: 0.3936 - accuracy: 0.8440 - val_loss: 0.1007 - val_accuracy: 0.9669

Save The Model

```
In [97]: #Save the model  
model.save('forest1.h5')
```

Predictions

```
In [25]: #import load_model from keras.model  
from keras.models import load_model
```

```
In [26]: #import image class from keras  
from keras.preprocessing import image
```

```
In [27]: #import numpy  
import numpy as np
```

```
In [28]: #import cv2  
import cv2
```

```
In [29]: #Load the saved model  
model = load_model('forest1.h5')
```

```
In [39]: #give any random image path  
img = image.load_img(r'_101542074_gettyimages_956391468.jpg')  
x = image.img_to_array(img)  
res=cv2.resize(x,dsize=(128,128),interpolation=cv2.INTER_CUBIC)
```

```
In [40]: # expand the image shape  
x=np.expand_dims(res,axis=0)
```

```
In [42]: pred=model.predict(x)
```

```
In [43]: pred
```

```
Out[43]: array([[0.]], dtype=float32)
```