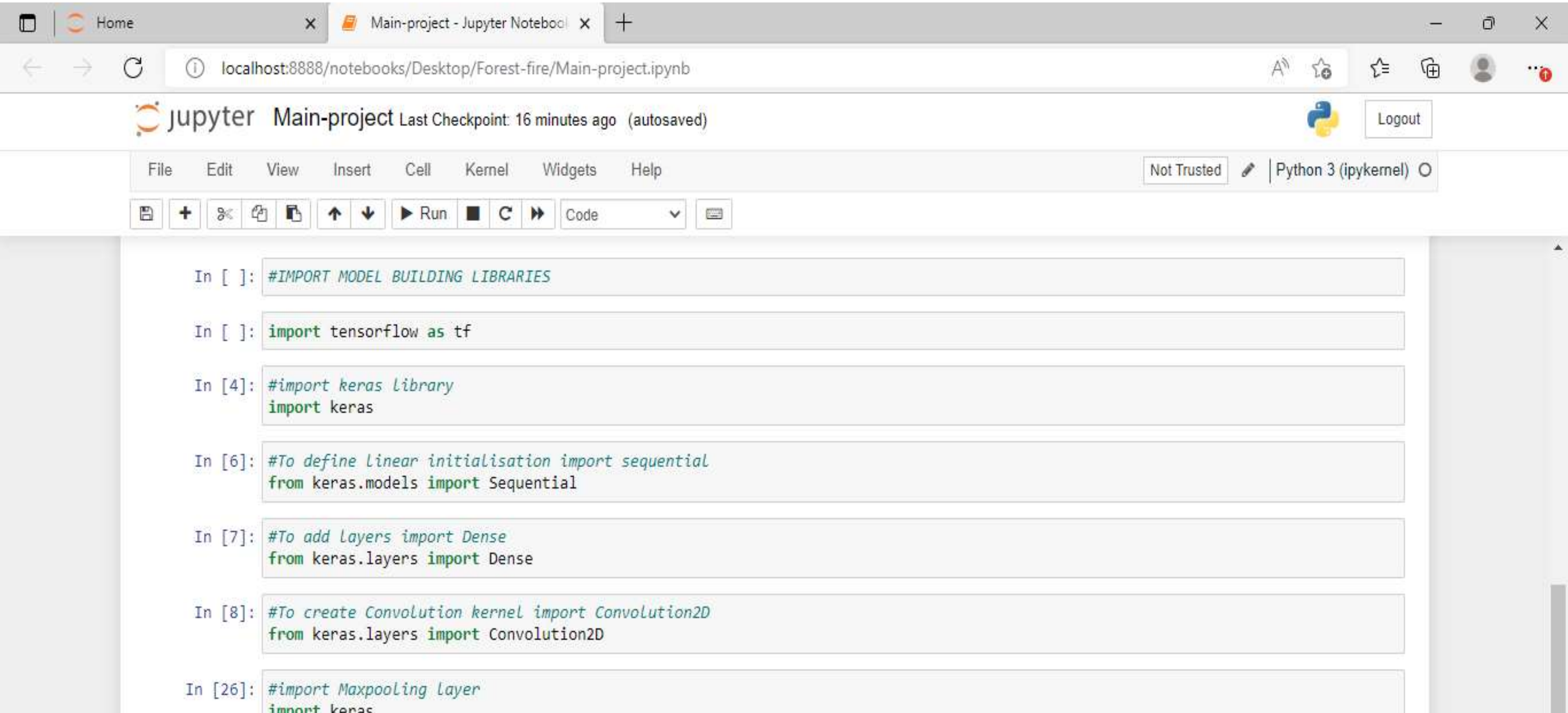
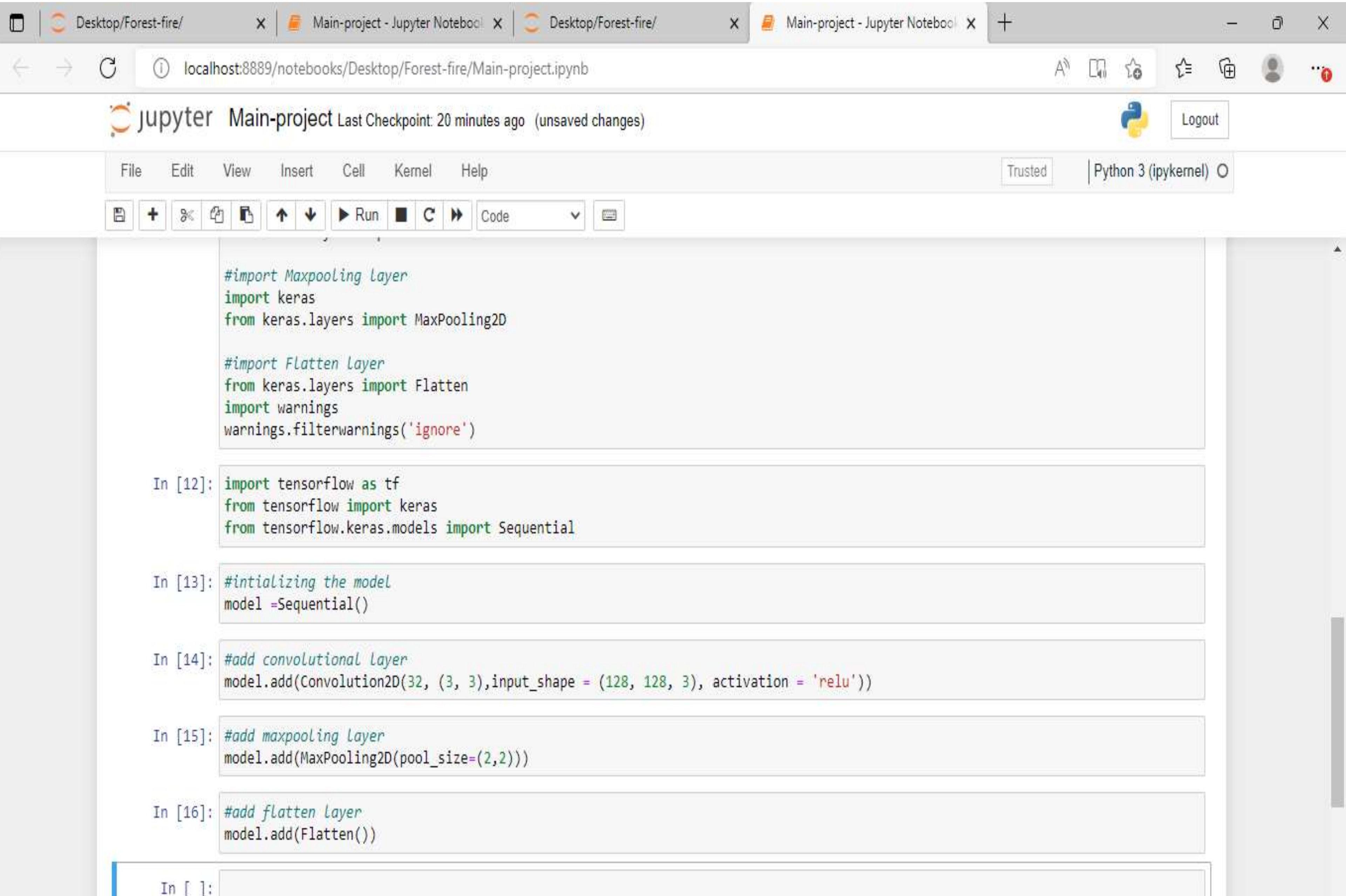
Model Building

Importing The Model Building Libraries


Home
Main-project - Jupyter Noteboo
localhost:8888/notebooks/Desktop/Forest-fire/Main-project.ipynb
jupyter
Main-project Last Checkpoint: 16 minutes ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
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


Desktop/Forest-fire/
Main-project - Jupyter Noteboo
Desktop/Forest-fire/
Main-project - Jupyter Noteboo
localhost:8889/notebooks/Desktop/Forest-fire/Main-project.ipynb
jupyter
Main-project Last Checkpoint: 20 minutes ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Help
Trusted
Python 3 (ipykernel) ○
Run
Code

```
#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]: #intializing the model
         model =Sequential()
```

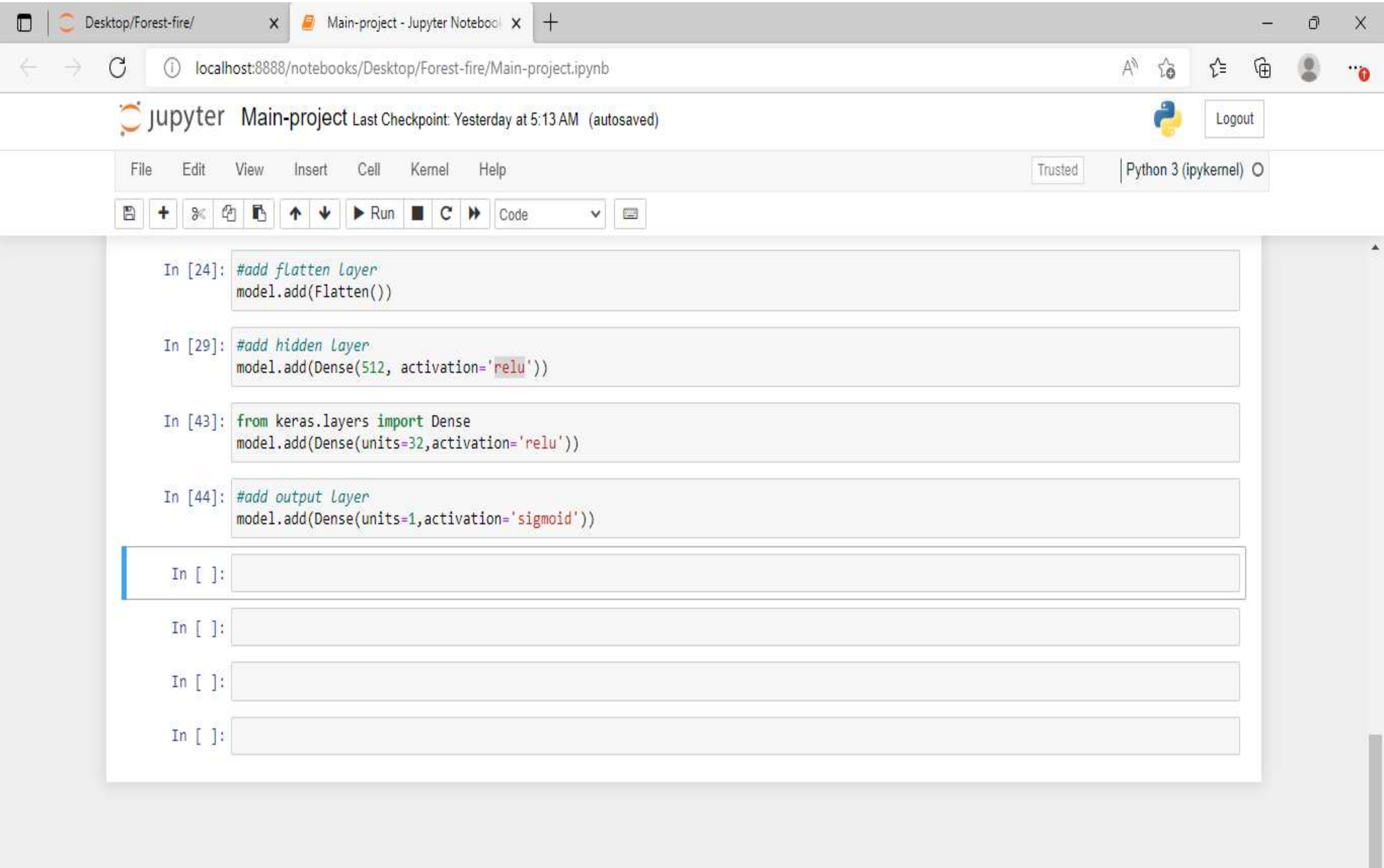
```
In [14]: #add convolutional layer
         model.add(Convolution2D(32, (3, 3),input_shape = (128, 128, 3), activation = 'relu'))
```

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

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

```
In [ ]:
```

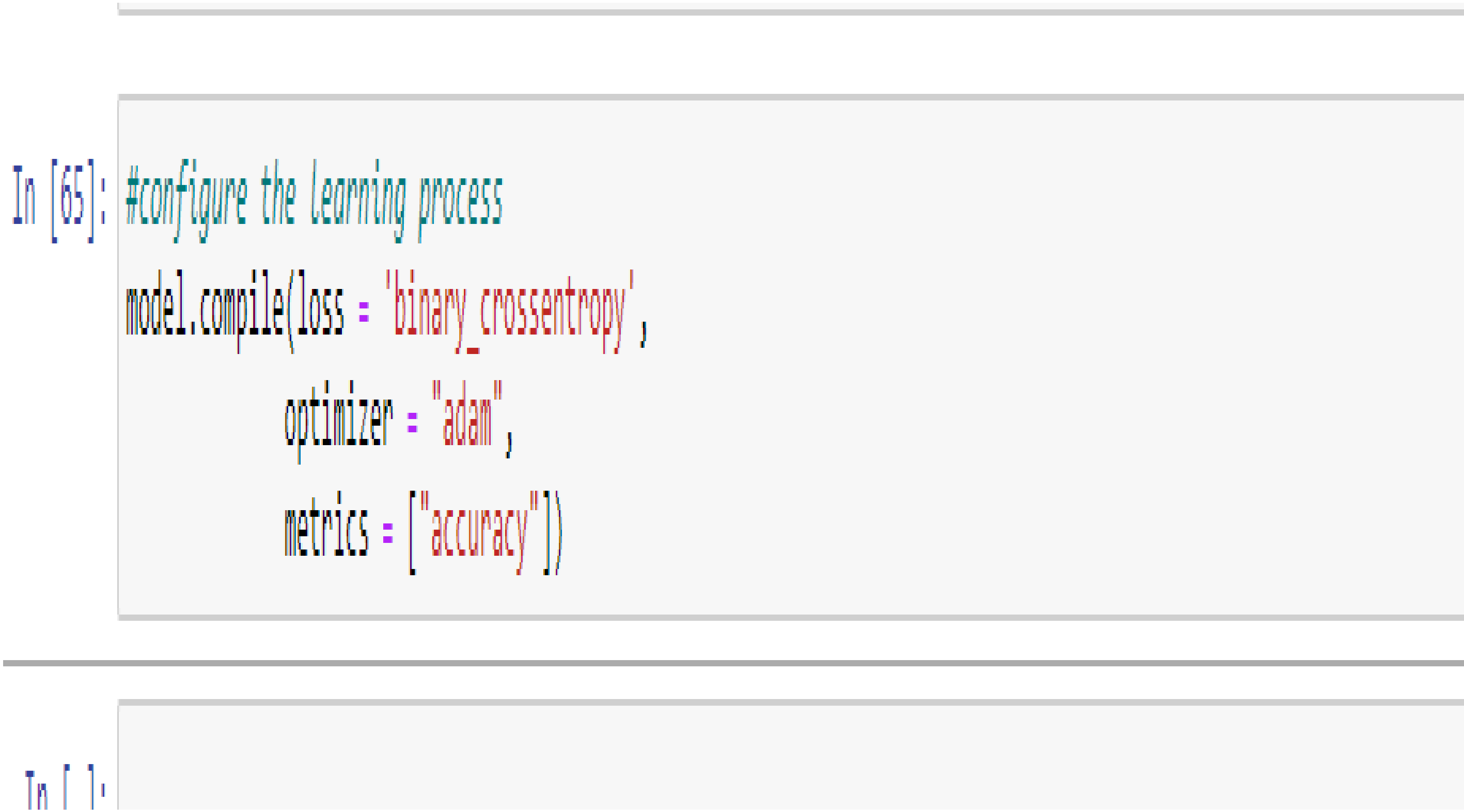
Adding Dense Layers



Desktop/Forest-fire/
Main-project - Jupyter Noteboo
localhost:8888/notebooks/Desktop/Forest-fire/Main-project.ipynb
jupyter
Main-project Last Checkpoint: Yesterday at 5:13 AM (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Help
Trusted
Python 3 (ipykernel)
Run
Code

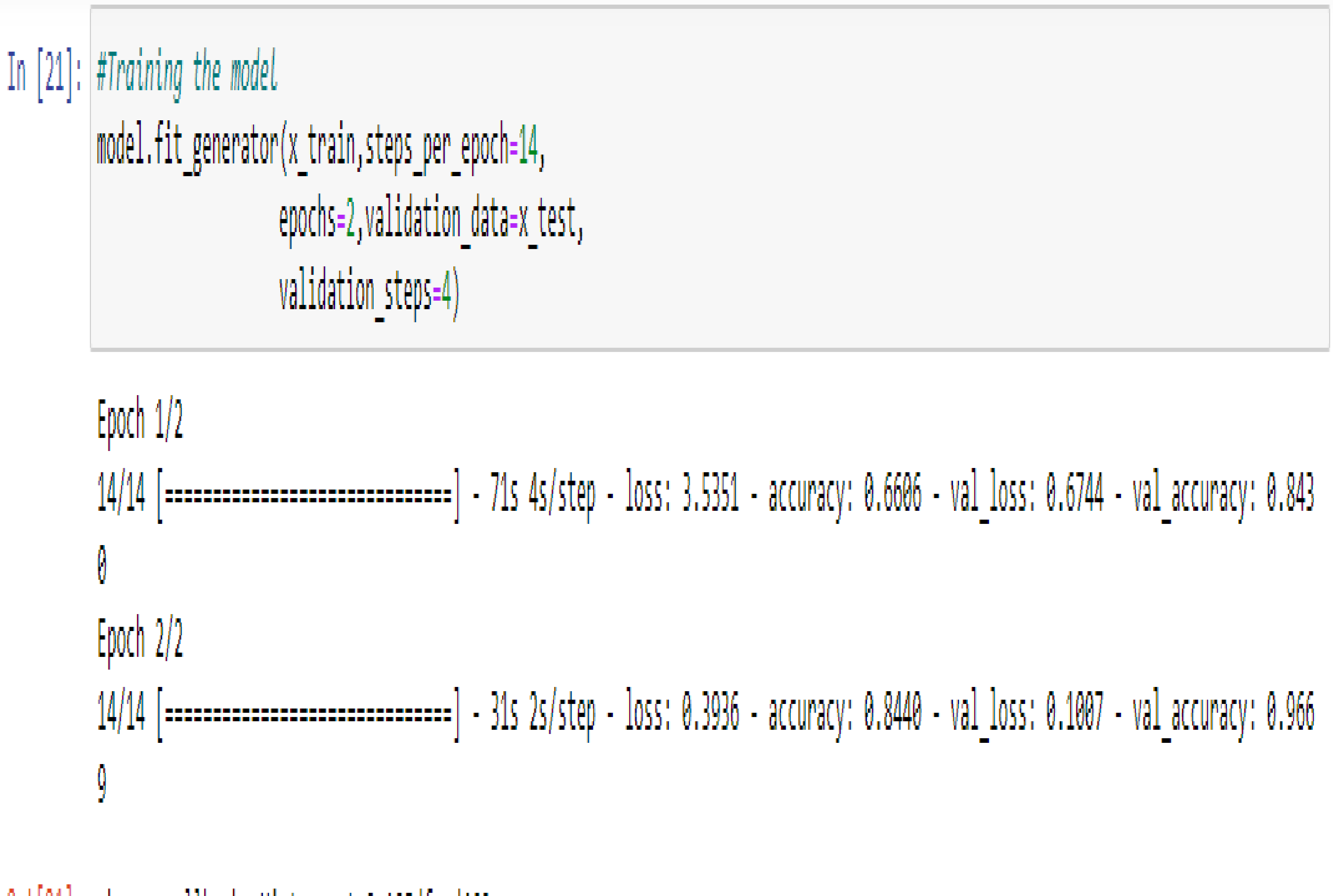
```
In [24]: #add flatten layer
         model.add(Flatten())

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"])

In [ ]:
```

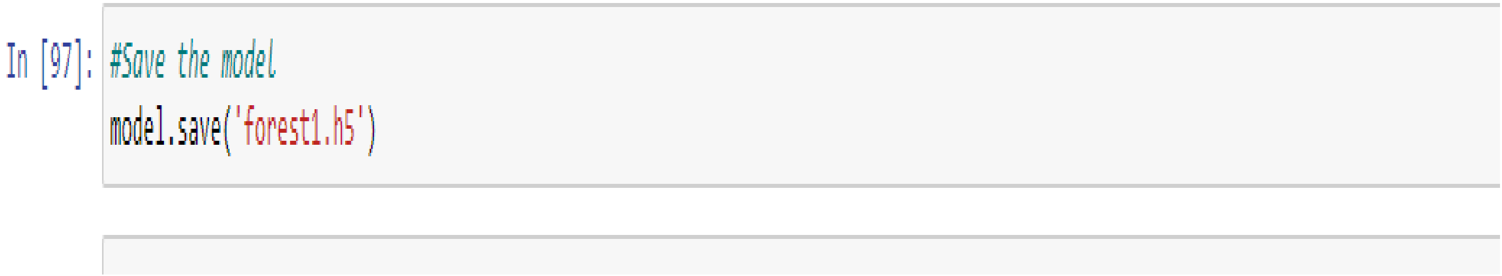
Training The Model



```
In [21]: #Training the model
         model.fit_generator(x_train,steps_per_epoch=14,
                             epochs=2,validation_data=x_test,
                             validation_steps=4)
```

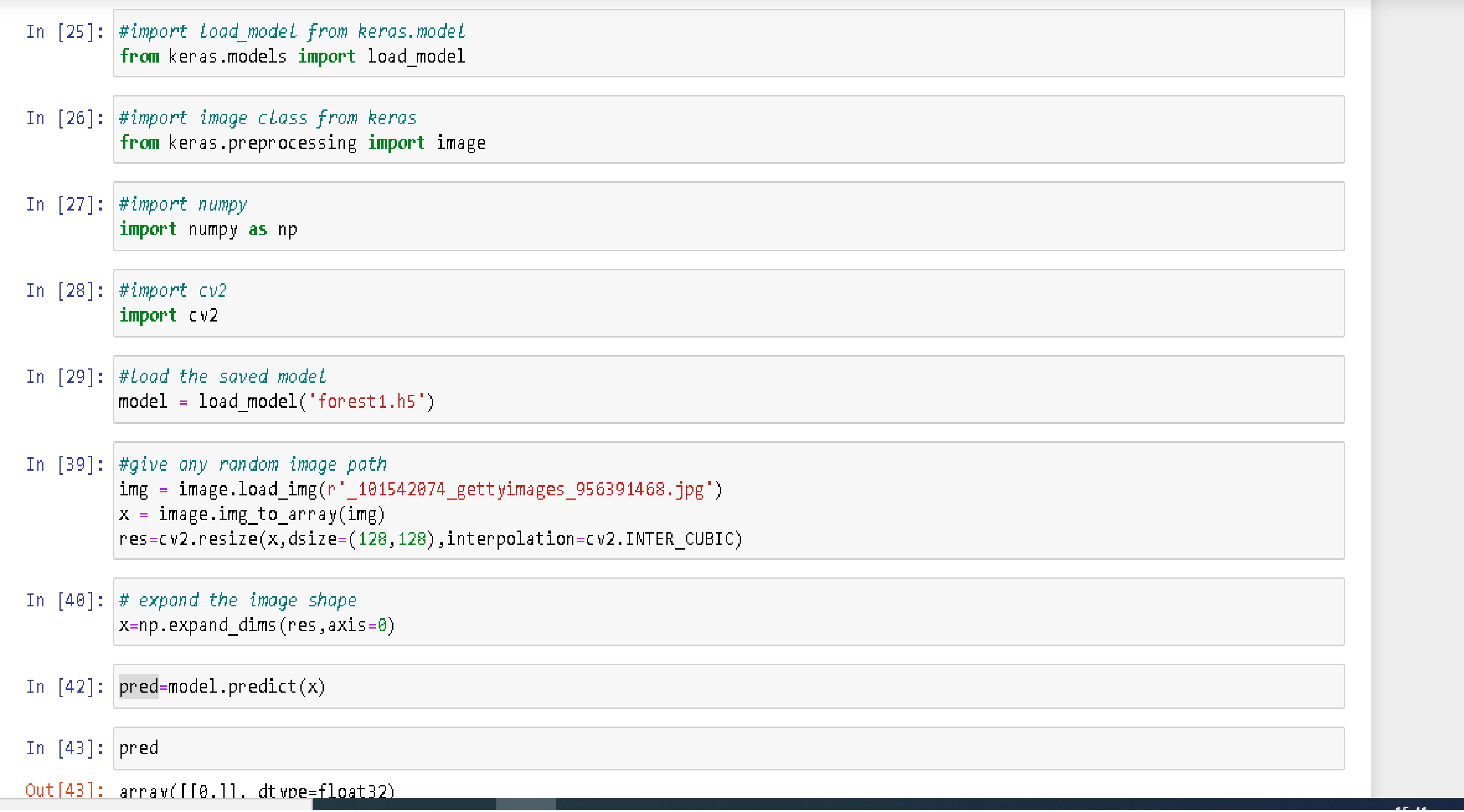
```
Epoch 1/2
14/14 [==============================] - 71s 4s/step - loss: 3.5351 - accuracy: 0.6606 - val_loss: 0.6744 - val_accuracy: 0.843
0
Epoch 2/2
14/14 [==============================] - 31s 2s/step - loss: 0.3936 - accuracy: 0.8440 - val_loss: 0.1007 - val_accuracy: 0.966
9
```

Save The Model



```
#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)
```