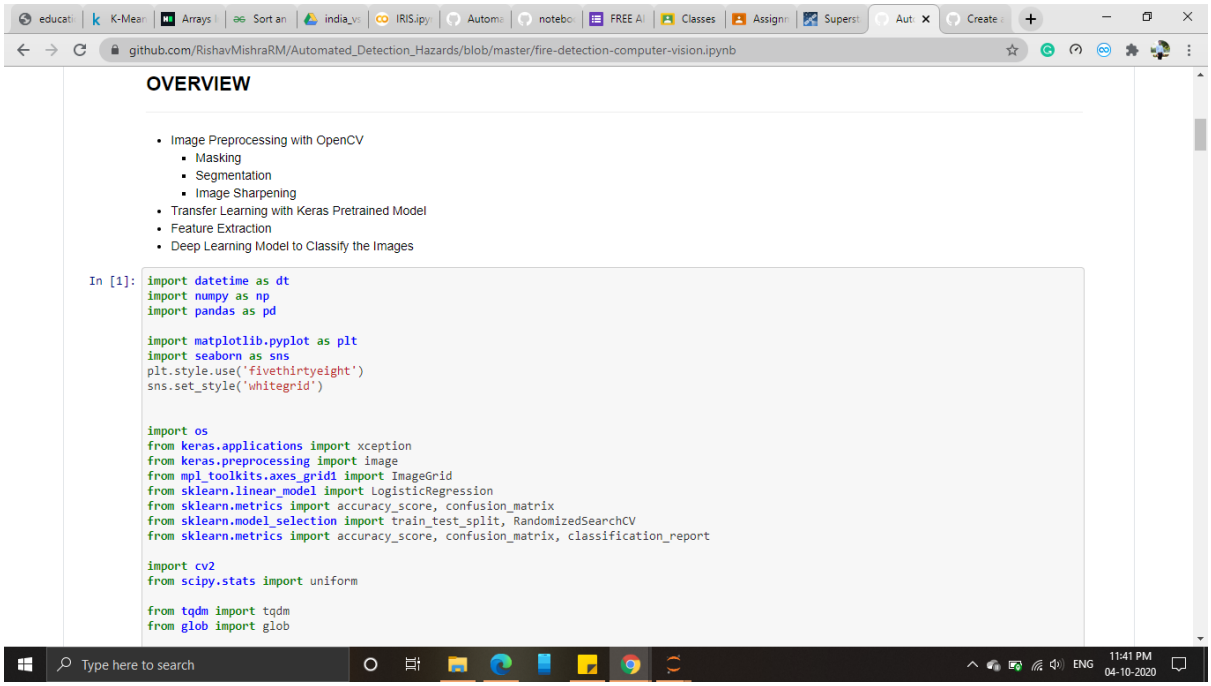
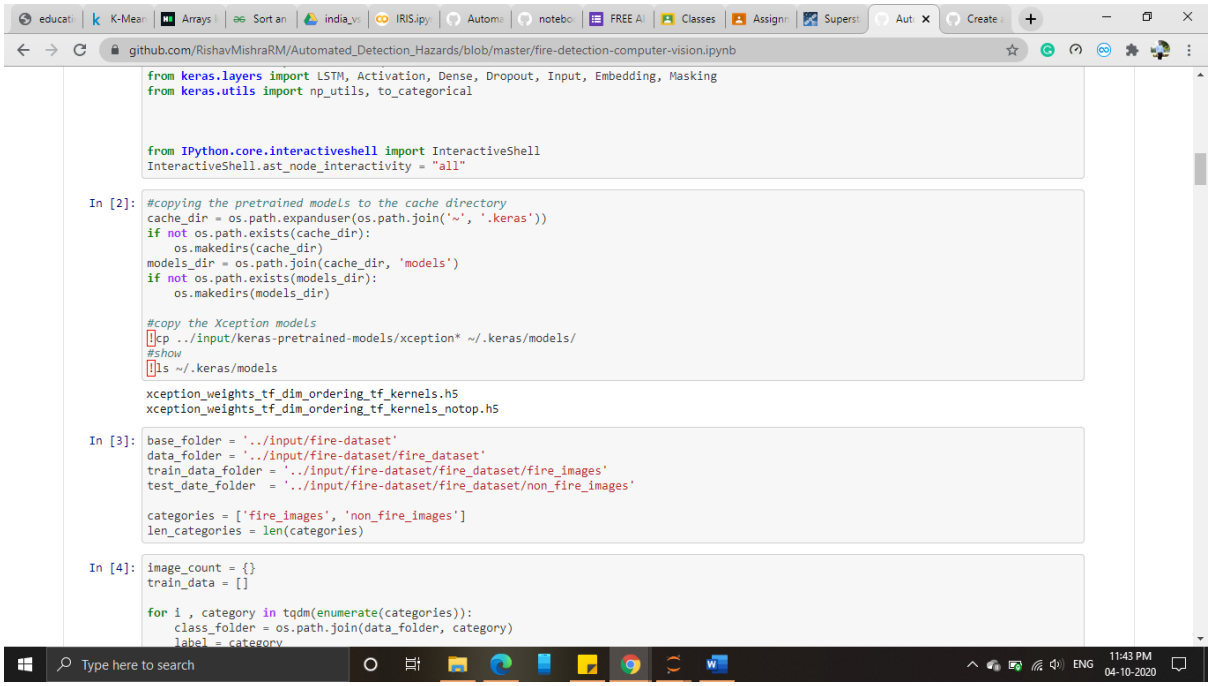
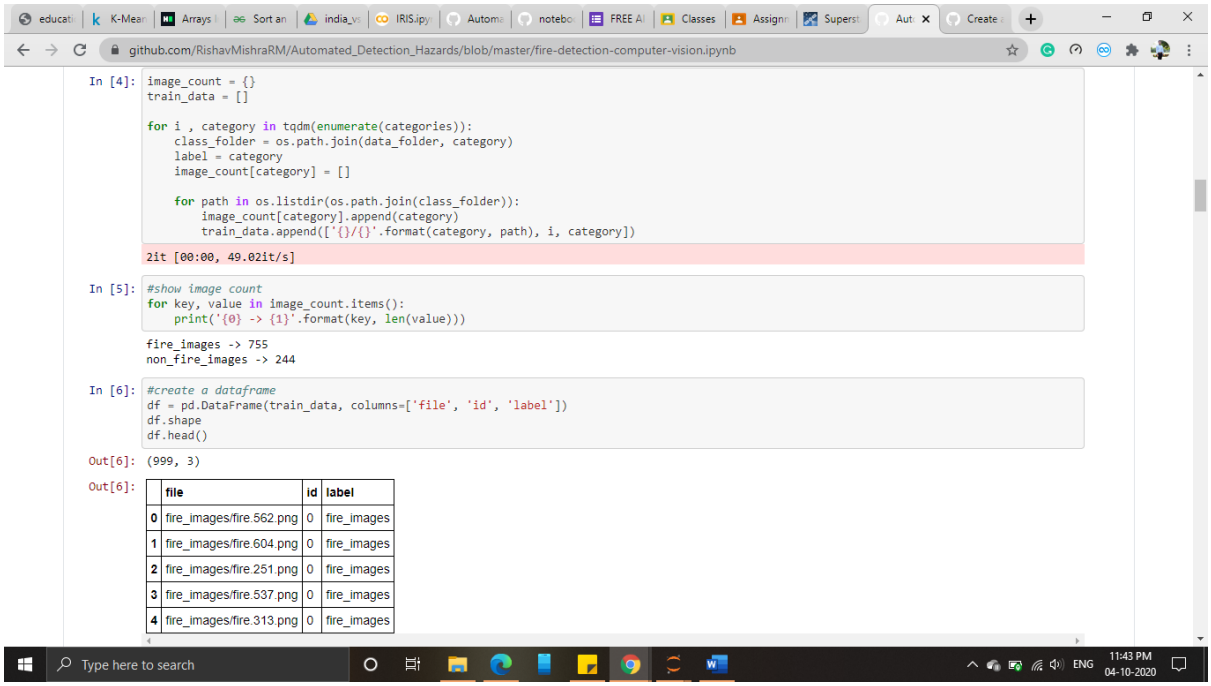
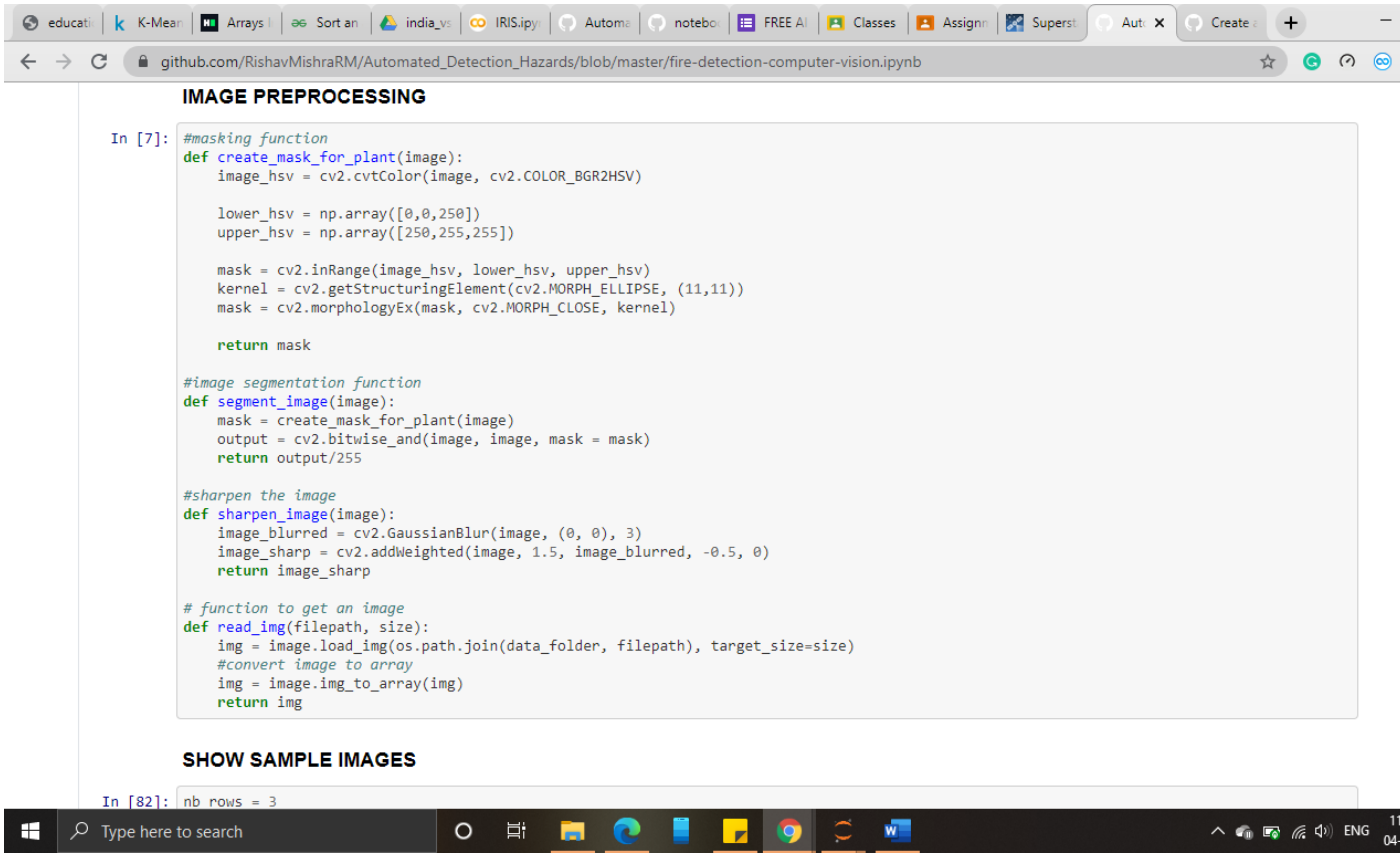

github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb
OVERVIEW
Image Preprocessing with OpenCV
Masking
Segmentation
Image Sharpening
Transfer Learning with Keras Pretrained Model
Feature Extraction
Deep Learning Model to Classify the Images
In [1]:
import datetime as dt
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')
sns.set_style('whitegrid')
import os
from keras.applications import xception
from keras.preprocessing import image
from mpl_toolkits.axes_grid1 import ImageGrid
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import cv2
from scipy.stats import uniform
from tqdm import tqdm
from glob import glob
Type here to search
ENG
11:41 PM
04-10-2020


github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb
from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding, Masking
from keras.utils import np_utils, to_categorical
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
In [2]:
#copying the pretrained models to the cache directory
cache_dir = os.path.expanduser(os.path.join('~', '.keras'))
if not os.path.exists(cache_dir):
os.makedirs(cache_dir)
models_dir = os.path.join(cache_dir, 'models')
if not os.path.exists(models_dir):
os.makedirs(models_dir)
#copy the Xception models
!cp ../input/keras-pretrained-models/xception* ~/.keras/models/
#show
!ls ~/.keras/models
xception_weights_tf_dim_ordering_tf_kernels.h5
xception_weights_tf_dim_ordering_tf_kernels_notop.h5
In [3]:
base_folder = '../input/fire-dataset'
data_folder = '../input/fire-dataset/fire_dataset'
train_data_folder = '../input/fire-dataset/fire_dataset/fire_images'
test_date_folder = '../input/fire-dataset/fire_dataset/non_fire_images'
categories = ['fire_images', 'non_fire_images']
len_categories = len(categories)
In [4]:
image_count = {}
train_data = []
for i , category in tqdm(enumerate(categories)):
class_folder = os.path.join(data_folder, category)
label = category
Type here to search
ENG
11:43 PM
04-10-2020


github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb
In [4]:
image_count = {}
train_data = []
for i , category in tqdm(enumerate(categories)):
class_folder = os.path.join(data_folder, category)
label = category
image_count[category] = []
for path in os.listdir(os.path.join(class_folder)):
image_count[category].append(category)
train_data.append(['{}/{}'.format(category, path), i, category])
2it [00:00, 49.02it/s]
In [5]:
#show image count
for key, value in image_count.items():
print('{0} -> {1}'.format(key, len(value)))
fire_images -> 755
non_fire_images -> 244
In [6]:
#create a dataframe
df = pd.DataFrame(train_data, columns=['file', 'id', 'label'])
df.shape
df.head()
Out[6]:
(999, 3)
Out[6]:
file
id
label
0
fire_images/fire.562.png
0
fire_images
1
fire_images/fire.604.png
0
fire_images
2
fire_images/fire.251.png
0
fire_images
3
fire_images/fire.537.png
0
fire_images
4
fire_images/fire.313.png
0
fire_images
Type here to search
ENG
11:43 PM
04-10-2020


github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb
IMAGE PREPROCESSING
In [7]:
#masking function
def create_mask_for_plant(image):
image_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
lower_hsv = np.array([0,0,250])
upper_hsv = np.array([250,255,255])
mask = cv2.inRange(image_hsv, lower_hsv, upper_hsv)
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (11,11))
mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
return mask
#image segmentation function
def segment_image(image):
mask = create_mask_for_plant(image)
output = cv2.bitwise_and(image, image, mask = mask)
return output/255
#sharpen the image
def sharpen_image(image):
image_blurred = cv2.GaussianBlur(image, (0, 0), 3)
image_sharp = cv2.addWeighted(image, 1.5, image_blurred, -0.5, 0)
return image_sharp
# function to get an image
def read_img(filepath, size):
img = image.load_img(os.path.join(data_folder, filepath), target_size=size)
#convert image to array
img = image.img_to_array(img)
return img
SHOW SAMPLE IMAGES
In [82]:
nb_rows = 3
Type here to search
ENG

github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb

SHOW SAMPLE IMAGES

In [82]:
```
nb_rows = 3
nb_cols = 5
fig, axs = plt.subplots(nb_rows, nb_cols, figsize=(10, 5));
plt.suptitle('SAMPLE IMAGES');
for i in range(0, nb_rows):
    for j in range(0, nb_cols):
        axs[i, j].xaxis.set_ticklabels([]);
        axs[i, j].yaxis.set_ticklabels([]);
        axs[i, j].imshow((read_img(df['file'][np.random.randint(1000)], (255,255)))/255.);
plt.show();
```

SHOW SAMPLE PROCESSED IMAGE

In [83]:
```
#get an image
```

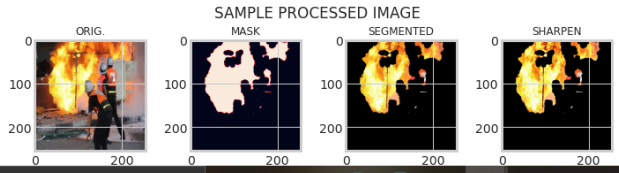
Type here to search

ENG 11:45 PM 04-10-2020

github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb

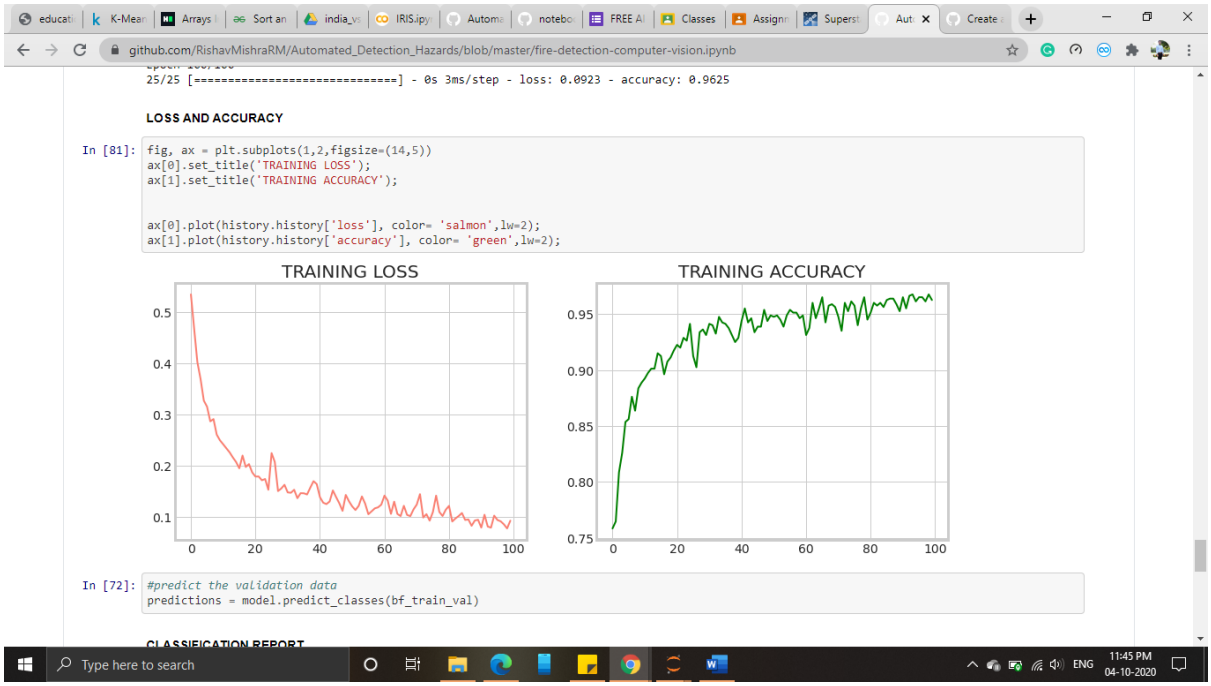
SHOW SAMPLE PROCESSED IMAGE

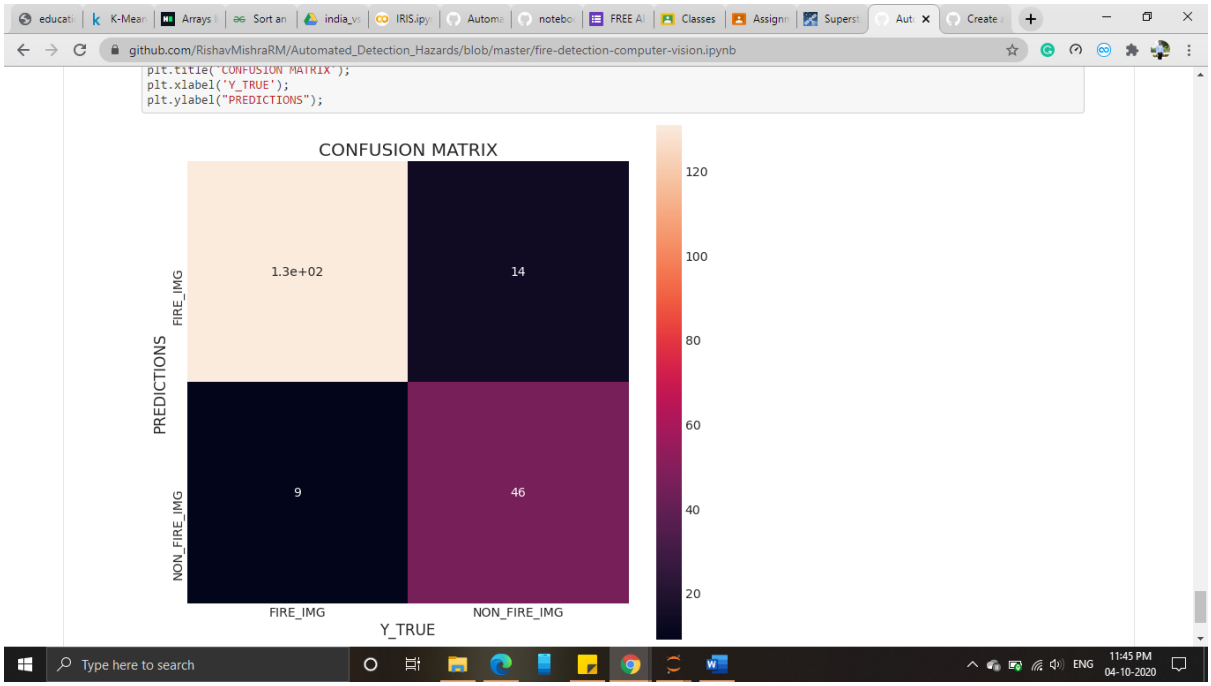
In [83]:
```
#get an image
img = read_img(df['file'][102],(255,255))
#mask
image_mask = create_mask_for_plant(img)
#segmentation
image_segmented = segment_image(img)
#sharpen the image
image_sharpen = sharpen_image(image_segmented)

fig, ax = plt.subplots(1, 4, figsize=(10, 5));
plt.suptitle('SAMPLE PROCESSED IMAGE', x=0.5, y=0.8)
plt.tight_layout(1)

ax[0].set_title('ORIG.', fontsize=12)
ax[1].set_title('MASK', fontsize=12)
ax[2].set_title('SEGMENTED', fontsize=12)
ax[3].set_title('SHARPEN', fontsize=12)


ax[0].imshow(img/255);
ax[1].imshow(image_mask);
ax[2].imshow(image_segmented);
ax[3].imshow(image_sharpen);
```



Type here to search

ENG 11:45 PM 04-10-2020


github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb
25/25 [==============================] - 0s 3ms/step - loss: 0.0923 - accuracy: 0.9625
LOSS AND ACCURACY
In [81]:
fig, ax = plt.subplots(1,2,figsize=(14,5))
ax[0].set_title('TRAINING LOSS');
ax[1].set_title('TRAINING ACCURACY');
ax[0].plot(history.history['loss'], color= 'salmon',lw=2);
ax[1].plot(history.history['accuracy'], color= 'green',lw=2);
TRAINING LOSS
TRAINING ACCURACY
In [72]:
#predict the validation data
predictions = model.predict_classes(bf_train_val)
Type here to search
ENG
11:45 PM
04-10-2020


github.com/RishavMishraRM/Automated_Detection_Hazards/blob/master/fire-detection-computer-vision.ipynb
plt.title('CONFUSION MATRIX');
plt.xlabel('Y_TRUE');
plt.ylabel("PREDICTIONS");
CONFUSION MATRIX
1.3e+02
14
9
46
FIRE_IMG
NON_FIRE_IMG
PREDICTIONS
FIRE_IMG
NON_FIRE_IMG
Y_TRUE
120
100
80
60
40
20
Type here to search
ENG
11:45 PM
04-10-2020