MODEL BUILDING

INITIALIZE THE MODEL

Date	04 November 2022
Team ID	PNT2022TMID13480
Project Name	Emerging methods for the early detection of forest fires

Initialzation:

```
add Codeadd Markdown
plt.style.use("dark_background"
) add Codeadd Markdown
GENERAL add Codeadd
Markdown
sns.countplot(Main_Train_Data["CATEGORY"])
plt.show() add Codeadd Markdown
Main_Train_Data['CATEGORY'].value_counts().plot.pie(figsize=(5,5))
plt.show() add Codeadd Markdown IMAGES add Codeadd
Markdown
figure = plt.figure(figsize=(10,10)) x =
cv2.imread(Main_Train_Data["PNG"][0])
plt.imshow(x) plt.xlabel(x.shape)
plt.title(Main_Train_Data["CATEGORY"][0]) add
Codeadd Markdown
figure = plt.figure(figsize=(10,10)) x =
cv2.imread(Main_Train_Data["PNG"][993])
plt.imshow(x) plt.xlabel(x.shape)
plt.title(Main_Train_Data["CATEGORY"][993])
add Codeadd Markdown
figure = plt.figure(figsize=(10,10)) x =
cv2.imread(Main_Train_Data["PNG"][20])
plt.imshow(x) plt.xlabel(x.shape)
plt.title(Main_Train_Data["CATEGORY"][20]) add
Codeadd Markdown
figure = plt.figure(figsize=(10,10)) x =
cv2.imread(Main_Train_Data["PNG"][48])
plt.imshow(x) plt.xlabel(x.shape)
plt.title(Main_Train_Data["CATEGORY"][48]) add
Codeadd Markdown
fig, axes = plt.subplots(nrows=5,
                   ncols=5,
figsize=(10,10),
                   subplot_kw={"xticks":[],"yticks":[]})

for i,ax in enumerate(axes.flat):
   ax.imshow(cv2.imread(Main_Train_Data["PNG"][i]))
ax.set_title(Main_Train_Data["CATEGORY"][i])
plt.tight_layout() plt.show() add Codeadd Markdown
fig, axes = plt.subplots(nrows=5,
                   ncols=5,
figsize=(10,10),
                   subplot_kw={"xticks":[],"yticks":[]})
```

```
for i,ax in enumerate(axes.flat):
   x = cv2.imread(Main_Train_Data["PNG"][i])
x = cv2.cvtColor(x,cv2.COLOR_RGB2BGR)
ax.imshow(x)
   ax.set_title(Main_Train_Data["CATEGORY"][i])
```

plt.tight_layout() plt.show() add Codeadd

Markdown DETERMINATION TRAIN AND TEST

DATA add Codeadd Markdown IMAGE

GENERATOR add Codeadd Markdown

```
Train_Generator = ImageDataGenerator(rescale=1./255,
                            shear_range=0.3,
zoom_range=0.2,
brightness_range=[0.2,0.9],
rotation_range=30,
horizontal_flip=True,
vertical_flip=True,
fill_mode="nearest",
validation_split=0.1)
```

add Codeadd Markdown

```
Test_Generator = ImageDataGenerator(rescale=1./255)
```

add Codeadd Markdown **SPLITTING TRAIN AND TEST** add

Codeadd Markdown

```
Train_Data,Test_Data =
train_test_split(Main_Train_Data,train_size=0.9,random_state=42,shuffle=Tru
e)
```

add Codeadd Markdown

```
print("TRAIN SHAPE: ",Train_Data.shape)
print("TEST SHAPE: ",Test_Data.shape)
add Codeadd Markdown
print(Train_Data.head(-1)) print("----
"*20) print(Test_Data.head(-1)) add
Codeadd Markdown
print(Test_Data["CATEGORY"].value_counts())
add Codeadd Markdown encode =
LabelEncoder() add Codeadd Markdown
For_Prediction_Class = encode.fit_transform(Test_Data["CATEGORY"]) add
Codeadd Markdown
How Generator Applied Image Look Like add
Codeadd Markdown
example_Image = Train_Data["PNG"][99]
Load_Image = image.load_img(example_Image,target_size=(200,200))
Array_Image = image.img_to_array(Load_Image)
Array_Image = Array_Image.reshape((1,) + Array_Image.shape)

i = 0 for batch in
Train_Generator.flow(Array_Image,batch_size=1):
plt.figure(i)
  IMG = plt.imshow(image.array_to_img(batch[0]))
i += 1    if i % 4 == 0:
    break plt.show()
add Codeadd Markdown
```

APPLYING GENERATOR AND TRANSFORMATION TO TENSOR add

Codeadd Markdown

```
Train_IMG_Set =
Train_Generator.flow_from_dataframe(dataframe=Train_Data,
                                    x_col="PNG",
y_col="CATEGORY",
color_mode="rgb",
class_mode="categorical",
batch_size=32,
subset="training")
```

add Codeadd Markdown

```
Validation_IMG_Set =
Train_Generator.flow_from_dataframe(dataframe=Train_Data,
                                    x_col="PNG",
y_col="CATEGORY",
color_mode="rgb",
class_mode="categorical",
batch_size=32,
subset="validation")
```

add Codeadd Markdown

```
Test_IMG_Set = Test_Generator.flow_from_dataframe(dataframe=Test_Data,
x_col="PNG",                                    y_col="CATEGORY",
color_mode="rgb",                                       class_mode="categorical",
batch_size=32)
```

add Codeadd Markdown CHECKING add

Codeadd Markdown for

```
data_batch,label_batch in Train_IMG_Set:
```

```
print("DATA SHAPE: ",data_batch.shape)
print("LABEL SHAPE: ",label_batch.shape)
break
add Codeadd Markdown for
data_batch,label_batch in Validation_IMG_Set:
   print("DATA SHAPE: ",data_batch.shape)
print("LABEL SHAPE: ",label_batch.shape)     break
add Codeadd Markdown for
data_batch,label_batch in Test_IMG_Set:
print("DATA SHAPE: ",data_batch.shape)
print("LABEL SHAPE: ",label_batch.shape)
break
add Codeadd Markdown print("TRAIN:
") print(Train_IMG_Set.class_indices)
print(Train_IMG_Set.classes[0:5])
print(Train_IMG_Set.image_shape)
print("---"*20) print("VALIDATION: ")
print(Validation_IMG_Set.class_indices)
print(Validation_IMG_Set.classes[0:5])
print(Validation_IMG_Set.image_shape)
print("---"*20) print("TEST: ")
print(Test_IMG_Set.class_indices)
print(Test_IMG_Set.classes[0:5])
print(Test_IMG_Set.image_shape)
```