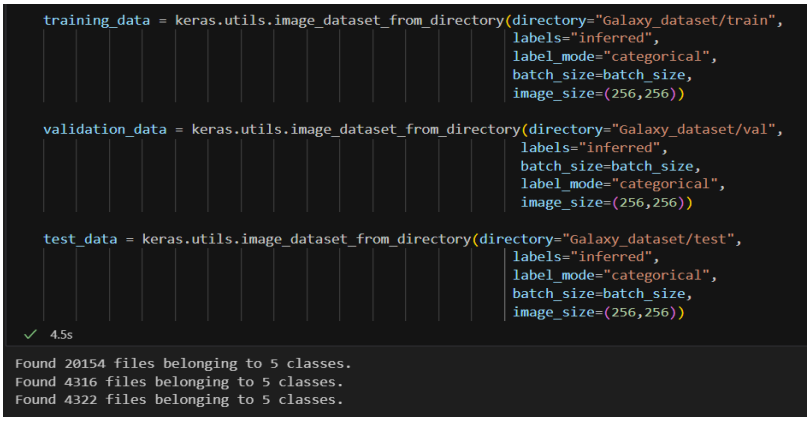
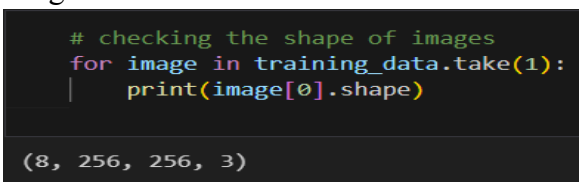
Data Collection and Preprocessing Phase

Date	16 June 2025
Team Lead Name	Jayanth Srinivas Bommisetty
Project Title	Sloan Digital Sky Survey (SDSS) galaxy classification using machine learning
Maximum Marks	6 Marks

Data Exploration and Preprocessing Template

The dataset was preprocessed by first employing ImageDataGenerator to perform real-time augmentation, enhancing the model's ability to generalize. All image pixel values were normalized to the 0–1 range to ensure consistent training behavior. The dataset was then split into training, validation, and test sets using the splitfolders library. Additionally, all images were resized to a uniform shape compatible with the CNN input requirements.

Section	Description
Data Overview	Dimension: Total:(28,792 images, 5 classes) Classified into train, test and val folders.  Image dimensions: 

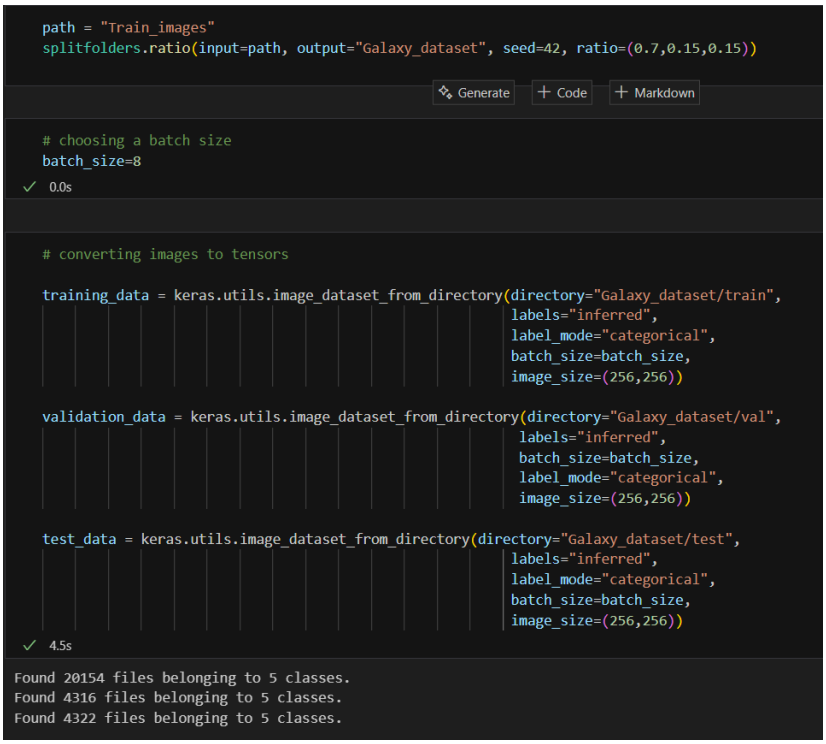
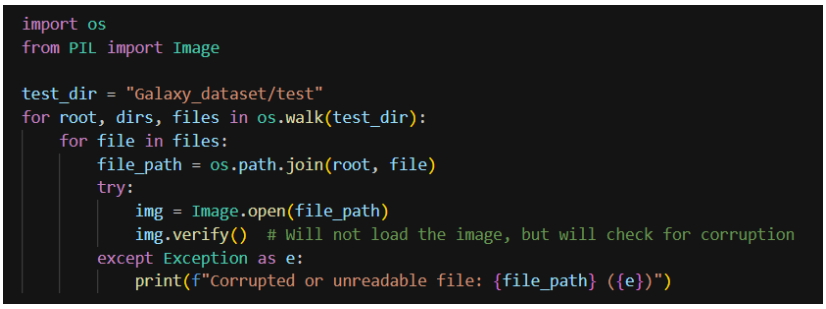
```
training_data = keras.utils.image_dataset_from_directory(directory="Galaxy_dataset/train",
                                                         labels="inferred",
                                                         label_mode="categorical",
                                                         batch_size=batch_size,
                                                         image_size=(256,256))

validation_data = keras.utils.image_dataset_from_directory(directory="Galaxy_dataset/val",
                                                           labels="inferred",
                                                           batch_size=batch_size,
                                                           label_mode="categorical",
                                                           image_size=(256,256))

test_data = keras.utils.image_dataset_from_directory(directory="Galaxy_dataset/test",
                                                     labels="inferred",
                                                     label_mode="categorical",
                                                     batch_size=batch_size,
                                                     image_size=(256,256))
```

✓ 4.5s

```
Found 20154 files belonging to 5 classes.
Found 4316 files belonging to 5 classes.
Found 4322 files belonging to 5 classes.
```

```
# checking the shape of images
for image in training_data.take(1):
    print(image[0].shape)
```

```
(8, 256, 256, 3)
```

Univariate Analysis	**Image processing**
Bivariate Analysis	**Image processing**
Multivariate Analysis	**Image processing**
Outliers and Anomalies	**Image processing**
Data Preprocessing Code Screenshots	
Loading Data	
Handling Missing Data	 Got some images corrupted issue and it is used to identify the location of the issue


```
import os
from PIL import Image

test_dir = "Galaxy_dataset/test"
for root, dirs, files in os.walk(test_dir):
    for file in files:
        file_path = os.path.join(root, file)
        try:
            img = Image.open(file_path)
            img.verify()  # Will not load the image, but will check for corruption
        except Exception as e:
            print(f"Corrupted or unreadable file: {file_path} ({e})")
```

Data Transformation	
Feature Engineering	
Save Processed Data	

```
training_data = keras.utils.image_dataset_from_directory(directory="Galaxy_dataset/train",
                                                         labels="inferred",
                                                         label_mode="categorical",
                                                         batch_size=batch_size,
                                                         image_size=(256,256))

validation_data = keras.utils.image_dataset_from_directory(directory="Galaxy_dataset/val",
                                                           labels="inferred",
                                                           batch_size=batch_size,
                                                           label_mode="categorical",
                                                           image_size=(256,256))

test_data = keras.utils.image_dataset_from_directory(directory="Galaxy_dataset/test",
                                                     labels="inferred",
                                                     label_mode="categorical",
                                                     batch_size=batch_size,
                                                     image_size=(256,256))
```

```
CNN_one = keras.models.Sequential()


# first conv and max layer + dropout layer to prevent overfitting
CNN_one.add(Conv2D(filters=32, kernel_size=3, activation="relu", input_shape=[256,256,3]))
CNN_one.add(MaxPool2D(pool_size=2, strides=2))
CNN_one.add(BatchNormalization())
CNN_one.add(Dropout(0.5))

# second conv and max layer + dropout layer
CNN_one.add(Conv2D(filters=65, kernel_size=3, activation="relu", kernel_regularizer="l2"))
CNN_one.add(MaxPool2D(pool_size=2, strides=2))
CNN_one.add(BatchNormalization())
CNN_one.add(Dropout(0.2))

# third conv and max layer + dropout layer
CNN_one.add(Conv2D(filters=128, kernel_size=3, activation="relu", kernel_regularizer="l2"))
CNN_one.add(MaxPool2D(pool_size=2, strides=2))
CNN_one.add(BatchNormalization())
CNN_one.add(Dropout(0.2))

# forth conv and max layer + dropout layer
CNN_one.add(Conv2D(filters=256, kernel_size=3, activation="relu", kernel_regularizer="l2"))
CNN_one.add(MaxPool2D(pool_size=2, strides=2))
CNN_one.add(BatchNormalization())
CNN_one.add(Dropout(0.2))

# flattening and dense layers
CNN_one.add(Flatten())
CNN_one.add(Dense(units=512, activation="relu"))
CNN_one.add(Dense(units=5, activation="softmax"))

CNN_one.summary()
```

```
# choosing the best model
best_model = CNN_one

# saving the best model
best_model.save("SDSSmodel.h5")
```