**ST1 – CAPSTONE PROJECT – U314854 Kanika Sharma**

**Report:**

***Fruits Dataset***

*Allocated Data: https://www.kaggle.com/datasets/shreyapmaher/fruits-dataset-images*

**Analysis/Design**

This project builds a deep learning model for fruit image classification using a Convolutional Neural Network (CNN). The model takes in input images of fruits and classifies them into different categories.

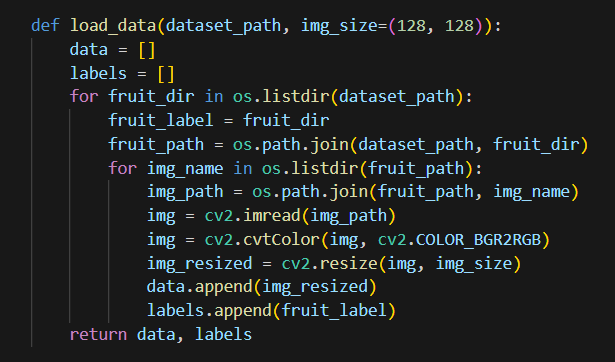
The implementation of this project involved multiple steps:

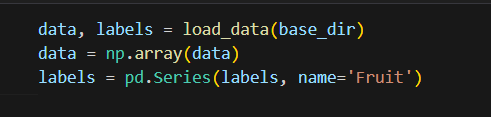
1. Loading and preprocessing the image data
2. Exploratory data analysis
3. Model implementation and training
4. Model performance evaluation
5. Deployment of the model using a Flask web application

**Algorithm Flowcharts/Pseudocode**

**Step 1: Load and Preprocess the Image Data**

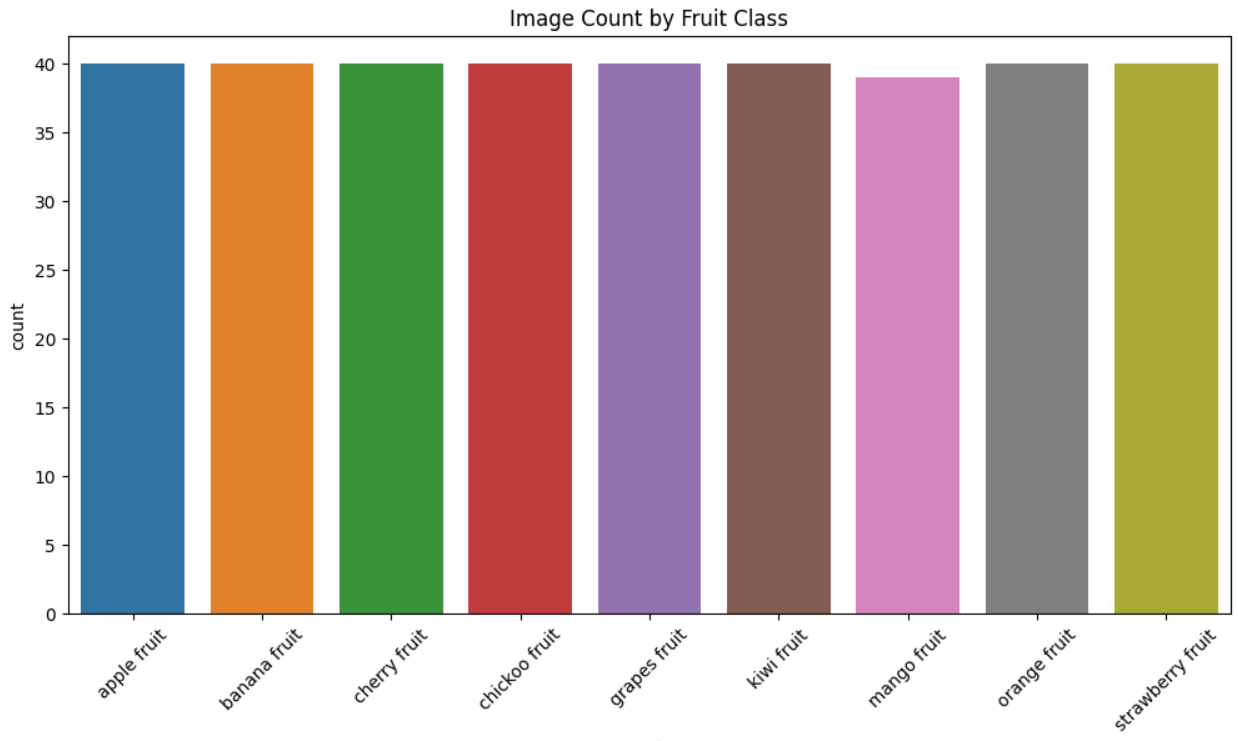
* Read the image data from the 'images' directory.
* Convert the images from BGR to RGB color space.
* Resize all the images to a standard size (128x128 pixels).
* Normalize the pixel values to fall between 0 and 1.

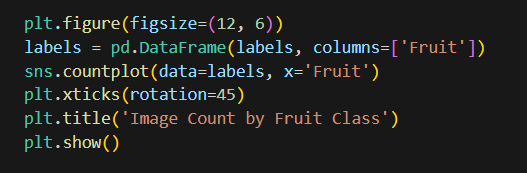




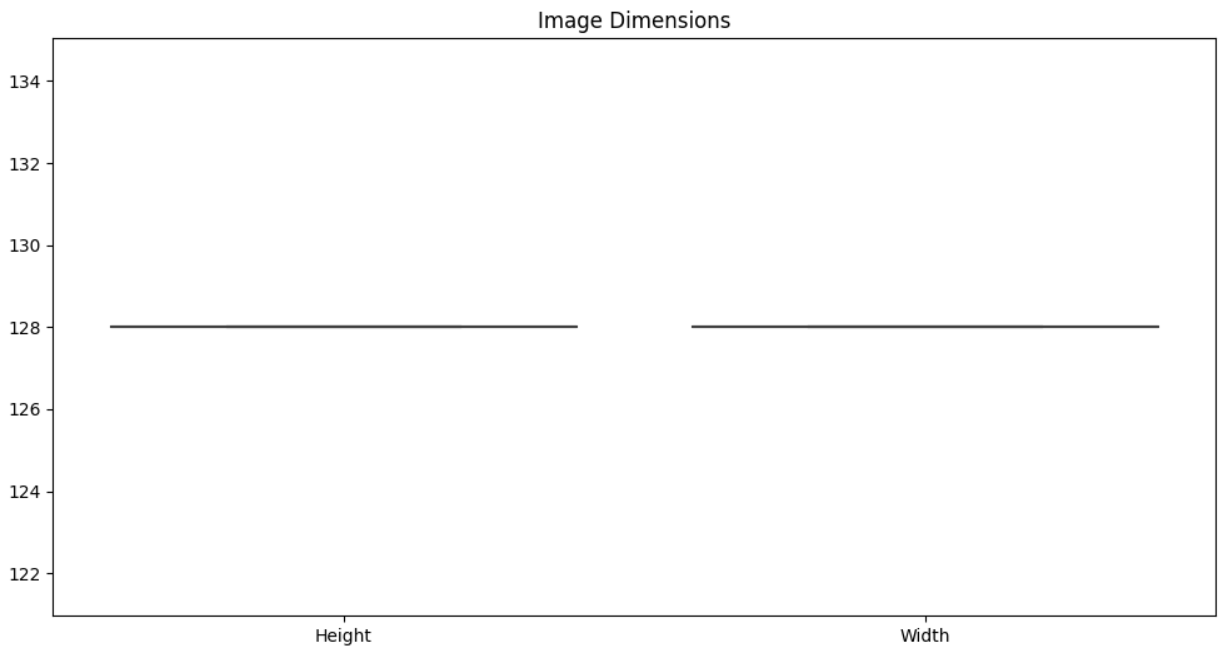
**Step 2: Exploratory Data Analysis**

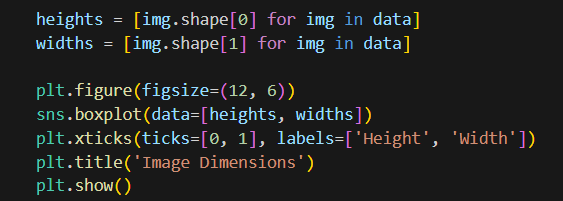
* **Question1 : What is the distribution of images among fruit classes?**

****

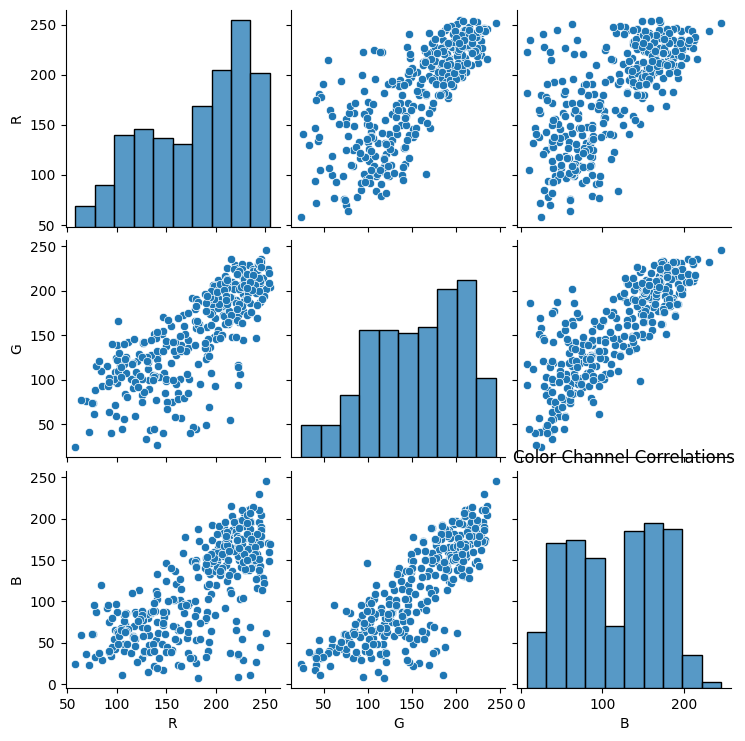


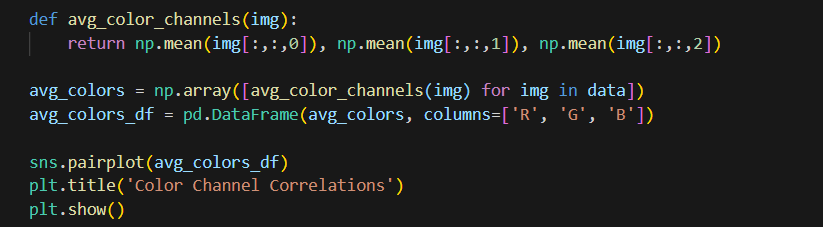
* **Question 2 : What is the average size of the Image?**

****

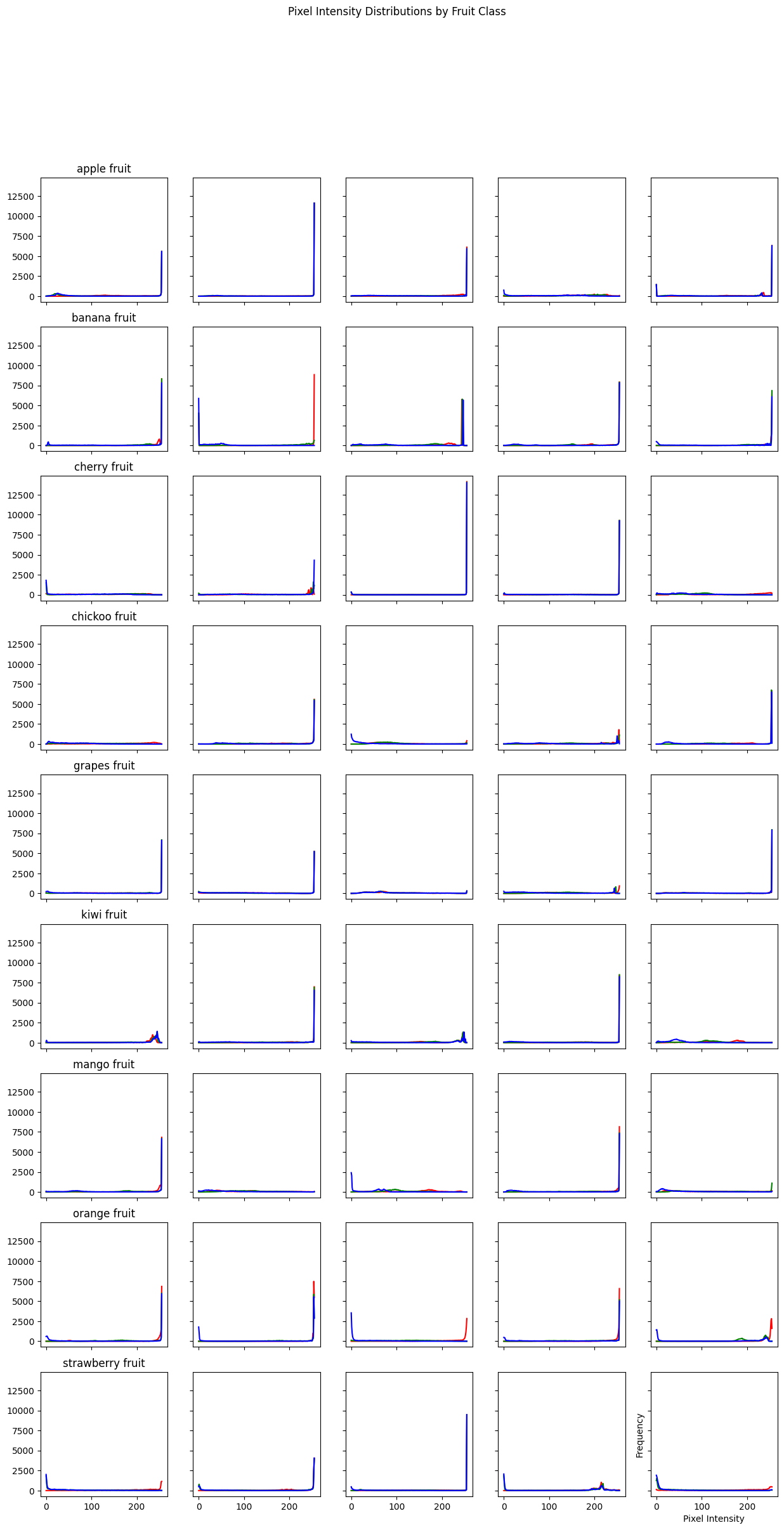
4

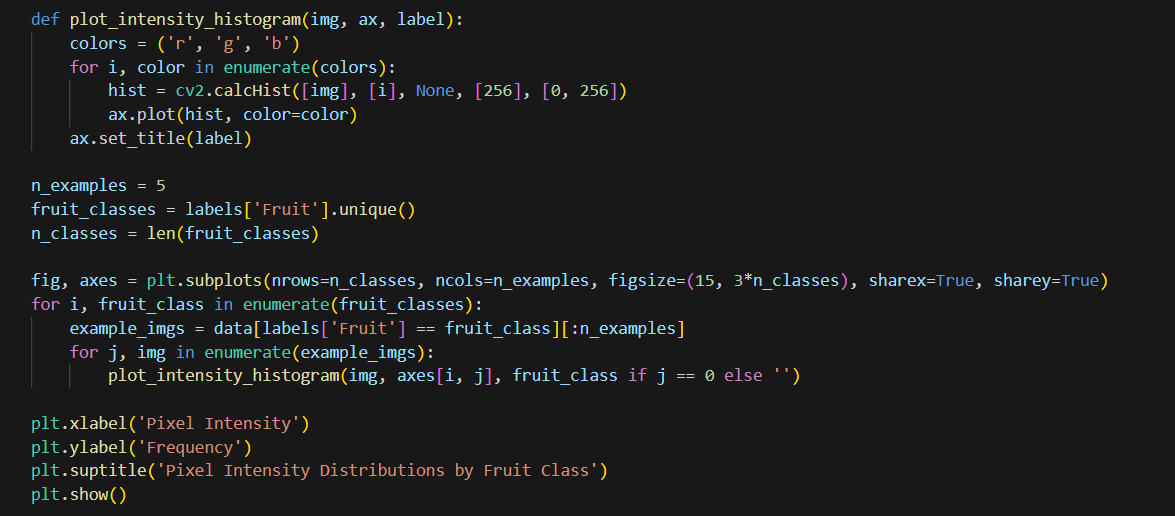
* **Question3 : Is there a correlation between different colour Channels?**

****

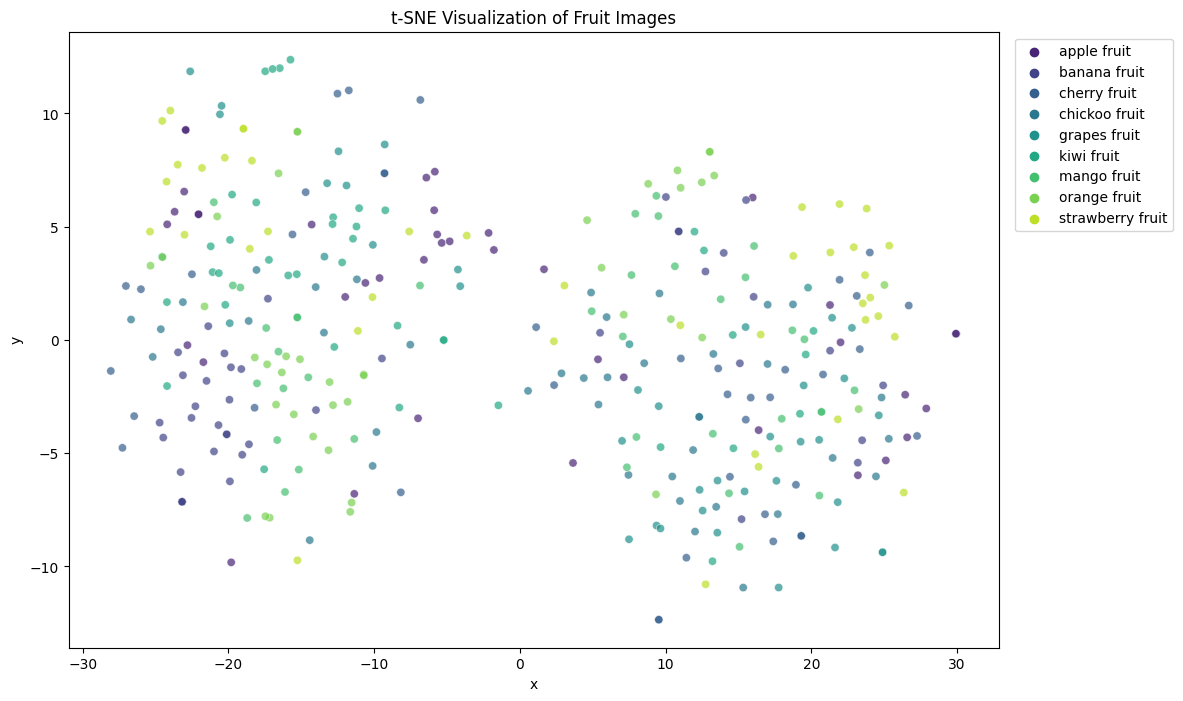


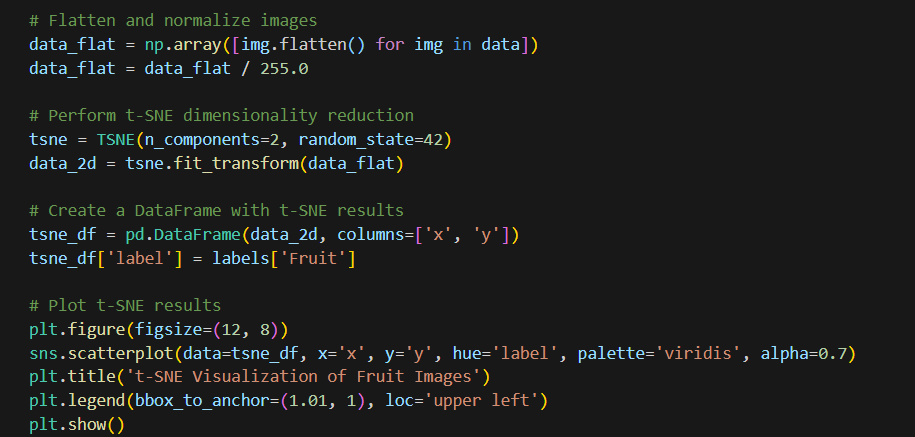
* **Question 4 : What is the pixel intensity and how it is distributed?**

****



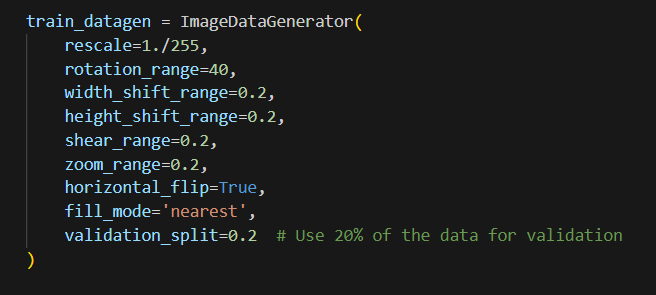
* **Question 5 : What is the distribution of classes ?**

****

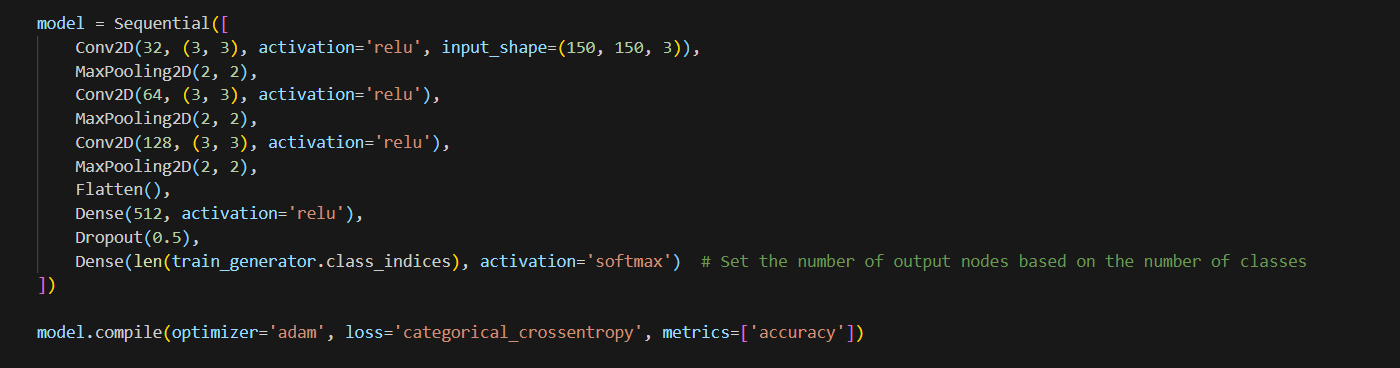


**Step 3: Model Implementation and Training**

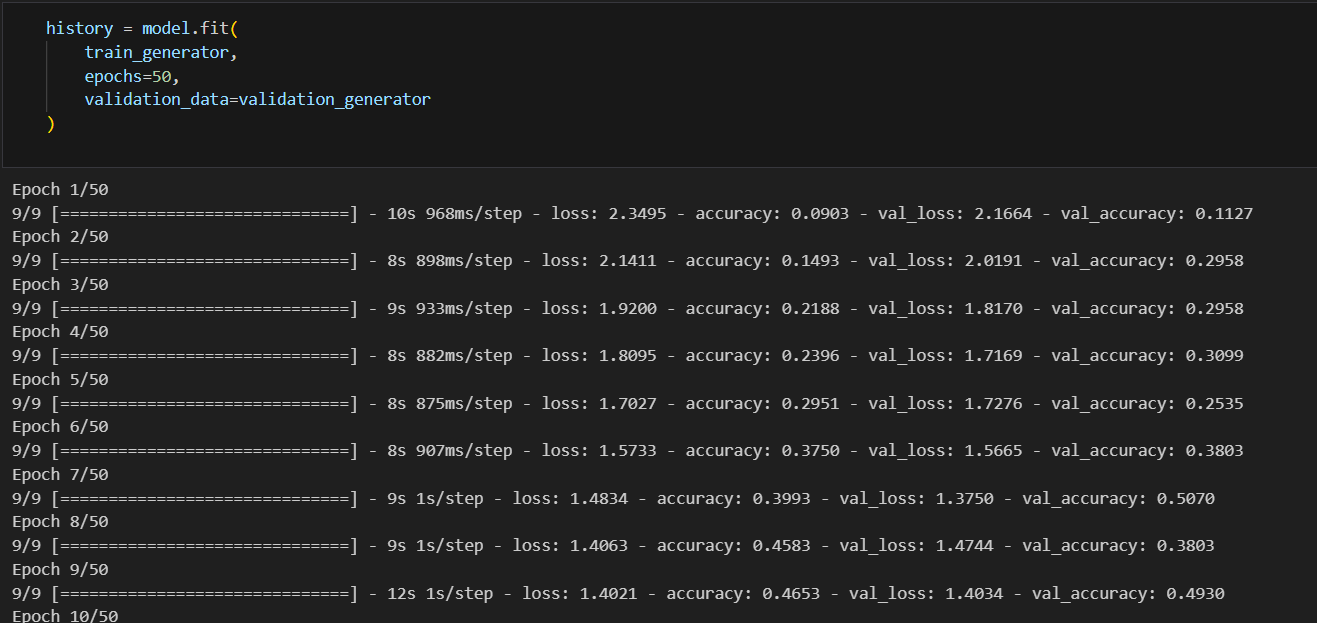
* Use TensorFlow's **ImageDataGenerator** to perform real-time data augmentation.



* Define a Sequential model with several Conv2D and MaxPooling2D layers, followed by a Flatten layer and two Dense layers.
* Compile the model with the 'adam' optimizer and 'categorical\_crossentropy' loss function.

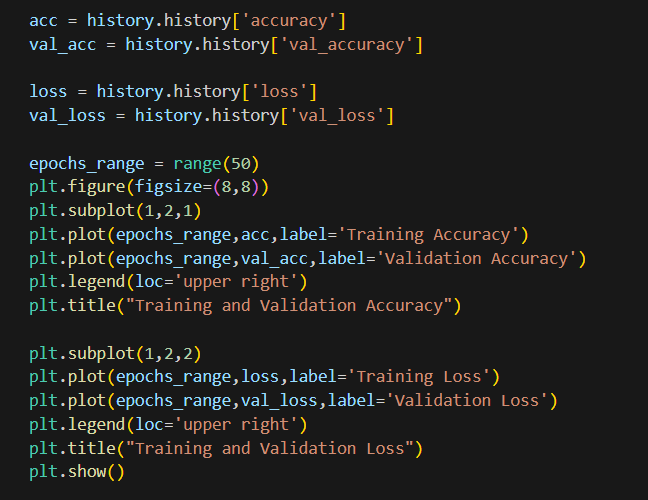


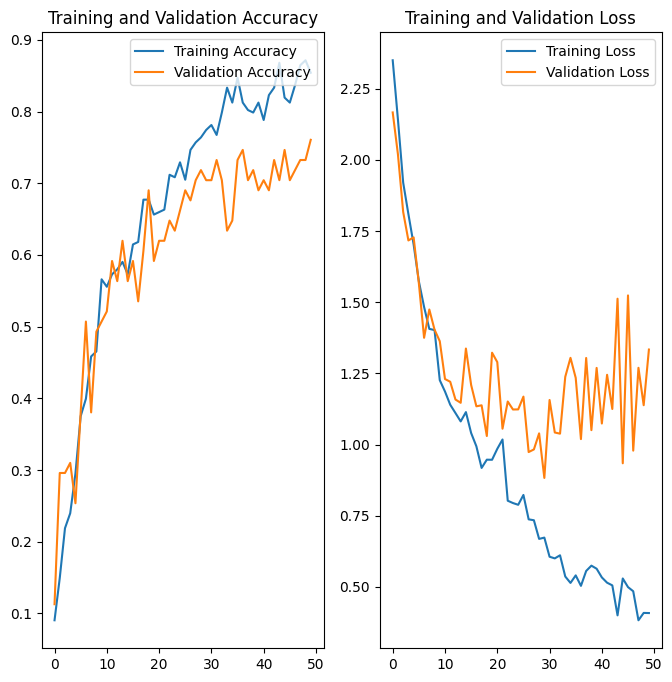
* Train the model using the augmented data, and save the trained model.



**Step 4: Model Performance Evaluation**

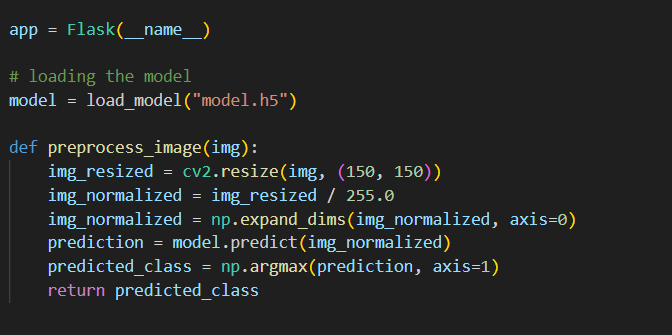
* Plot the training and validation accuracy and loss over the epochs to evaluate the model's performance.

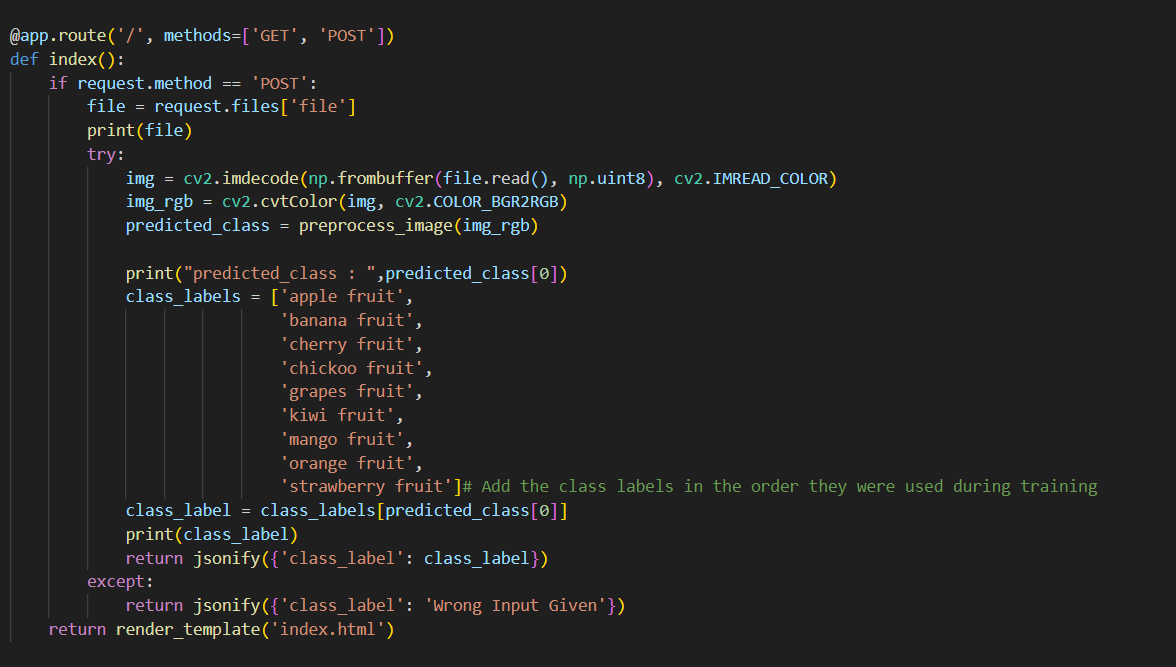




**Step 5: Deployment of the Model**

* Create a Flask web application.
* Load the trained model in the Flask app.
* Define a function to preprocess the uploaded image and make predictions using the trained model.
* Create an endpoint that accepts POST requests with an image file, processes the image, and returns the predicted fruit class.





**Code Listing**

The detailed Python code for the project is given above. The project uses libraries like OpenCV for image processing, TensorFlow for model creation and training, Seaborn and Matplotlib for data visualization, and Flask for web application creation.

**Code for Model Generation**

The model is trained using a deep learning CNN model which comprises of multiple convolutional layers followed by max-pooling layers, and finally the dense layers. The model takes in input images of size 150x150 with 3 color channels, and outputs a probability distribution over the different fruit classes. The model is trained for 50 epochs, and the 'adam' optimizer is used for training the model with the 'categorical\_crossentropy' loss function.

**Flask Code**

A Flask web application is created for users to upload fruit images and get predictions. The application loads the trained model and preprocesses the uploaded image before making predictions. It then returns the predicted fruit class. The web application can be started by running the Flask application, and users can interact with it via the provided HTML interface or directly by sending POST requests to the server.

**Evidence of Testing**

Test 1 : Input Image : Apple image

Test2 : Input Image : Banana image

Test 3 : Input Image : Docx format given

See the result below

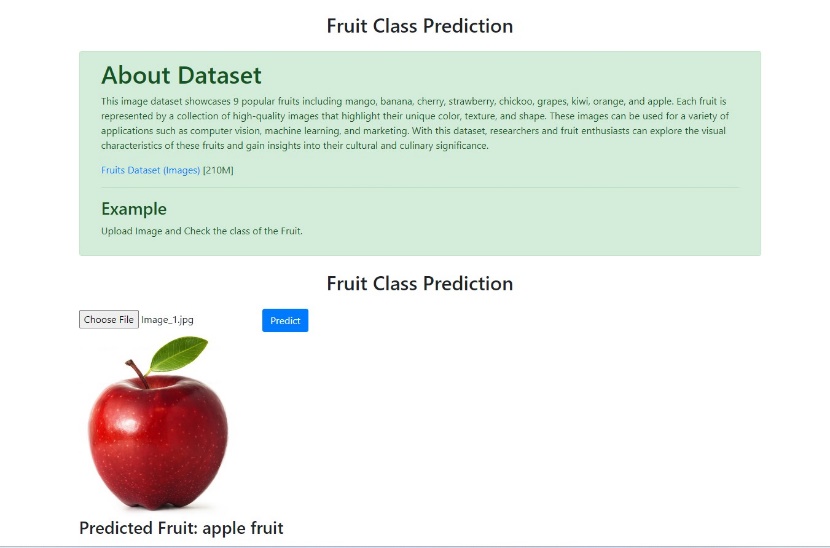
****

Figure Test1

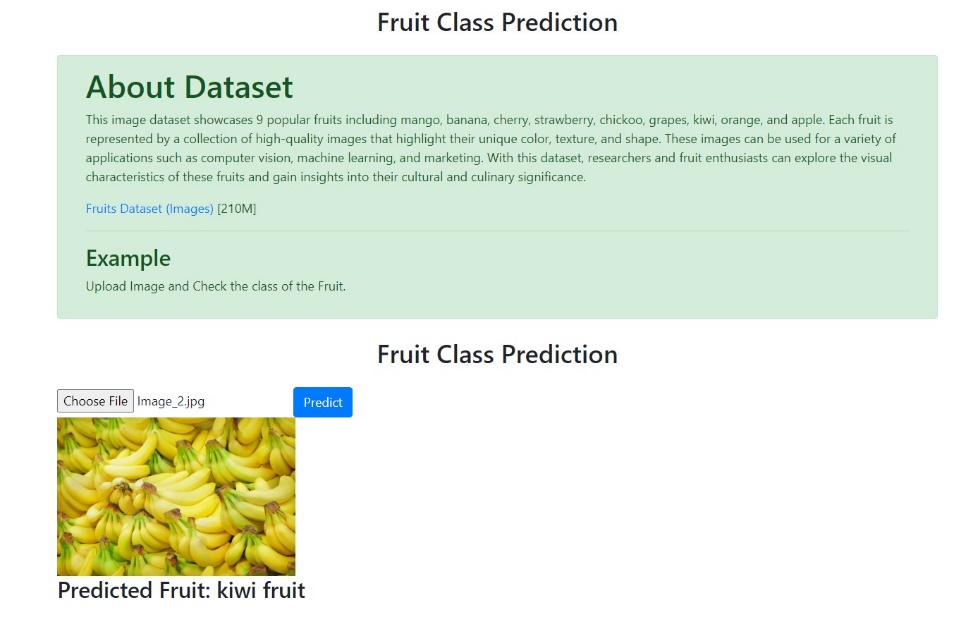


Figure : Test 2

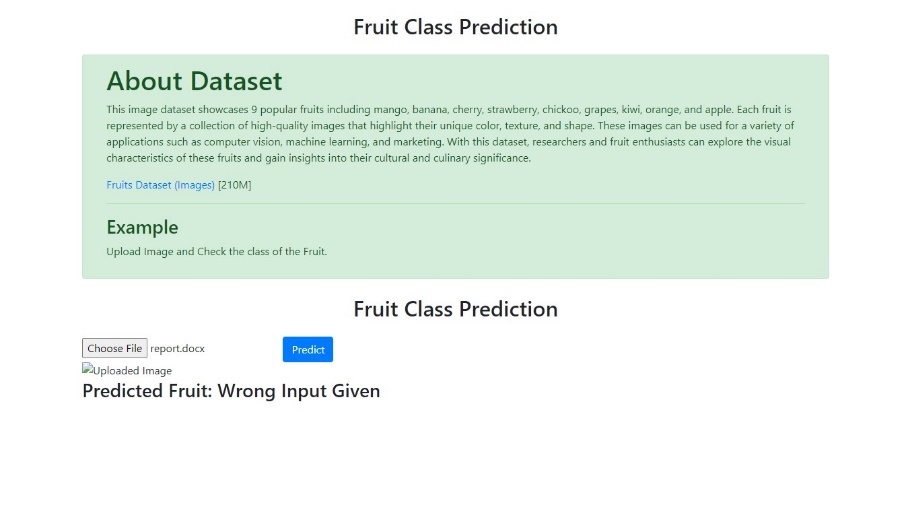


Figure Test 3

**Reflection**:

**What was learned**

The project provided a comprehensive learning experience in various aspects:

1. **Data Preprocessing and Analysis**: We learned how to load and preprocess image data to prepare it for a machine learning model. We also conducted exploratory data analysis, including visualizing the distribution of images among different fruit classes, plotting image sizes, and analyzing color channel correlations.
2. **Deep Learning Models**: The project offered hands-on experience in implementing and training a Convolutional Neural Network, a type of deep learning model particularly effective for image classification tasks.
3. **Model Evaluation**: We gained knowledge about monitoring training and validation accuracy and loss to understand how our model was performing and whether it was overfitting or underfitting.
4. **Model Deployment**: Implementing a Flask web application to use our trained model for making predictions on new data was an important learning aspect. This gave us insights into how machine learning models are deployed in real-world applications.

**Challenges faced**

Some of the challenges faced during this project included:

1. **Image Data Handling**: Dealing with image data was more complex compared to traditional structured data. The images had to be loaded and preprocessed correctly, including resizing and color space conversion, before they could be used in a model.
2. **Model Tuning**: Deciding on the right architecture for the CNN, such as the number of layers, types of layers, and their parameters, required experimentation.
3. **Overfitting**: As is common in deep learning models, our model was initially prone to overfitting. We overcame this using techniques like data augmentation and dropout.

**Future Improvements**

While the project achieved its objectives, there are several areas for potential improvement:

1. **Model Performance**: The model's performance could potentially be improved further by exploring different architectures, more advanced models like ResNets, or other techniques such as transfer learning.
2. **User Interface**: The user interface of the Flask application is quite basic and could be enhanced to provide a better user experience.
3. **Additional Features**: Currently, the application can only classify a single fruit in an image. In the future, we could extend the model to identify multiple fruits in a single image.

In conclusion, this project was a valuable learning experience in image classification using deep learning, and we look forward to leveraging these learnings in future projects.