**Fruit Type Classification**

**Using ML model implemented through Django API and .net + JQuery Client**

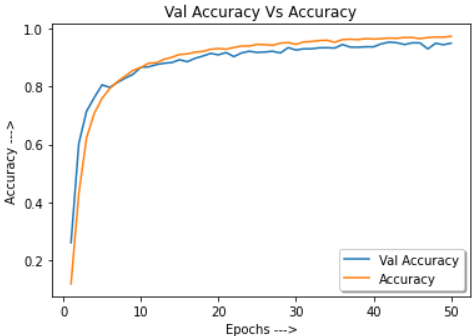
1. Introduction

This project uses image processing and categorical classification machine learning technique to classify the fruit type. This document describes different application created in this classification project.

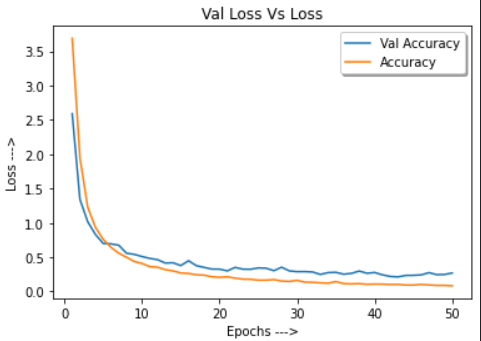
1. Model Generation
   1. Model Performance

Kaggle data set and python programming language with keras library has been used to generate the classifier model. Generated model is stored as .h5 file and implemented in Django API. At the end of this documentation, model generation code is attached.

* + 1. Val Accuracy Vs Accuracy



* + 1. Val Loss Vs Loss



* 1. Classification Matrix and report







1. API Application

The code snippet below is taken out the Django API view. We have a method FruitClassifierView that accepts the POST request from client. Client will submit image to this API. The image will then passed through the classifier model (.h5) and result is returned to the client.



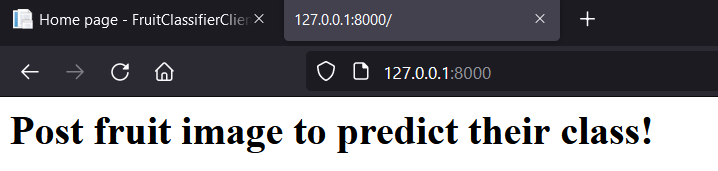
1. Client Application

The code snippet below shows the JQuery script to post image data to Django API and display the result to the client page.



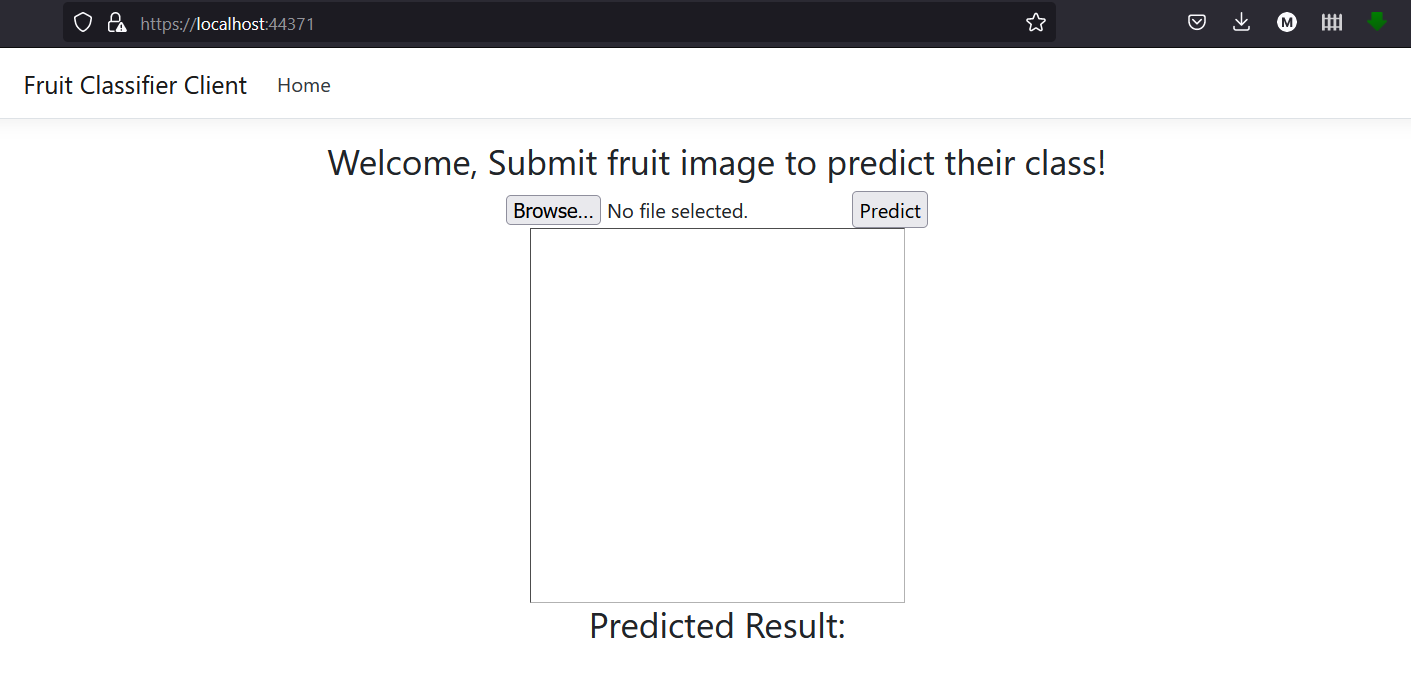
1. Classification Demo
   * 1. Classification API





* + 1. Client Start Page

The client home page has brows and predict button. Browse will allow selection of image and then only predict button is activated, through which, user will be able to request classification.



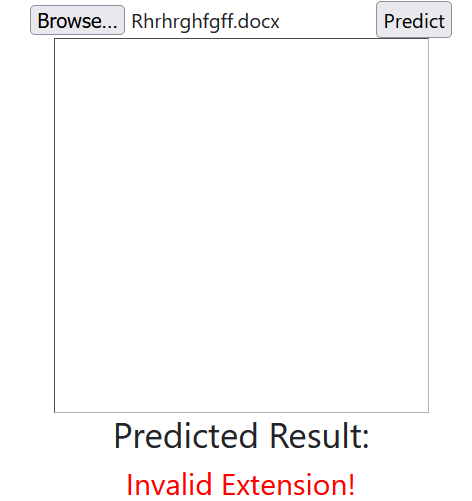
* + 1. Image Search

When user selects a valid image, image preview is displayed in the page and predict button is enabled as well.



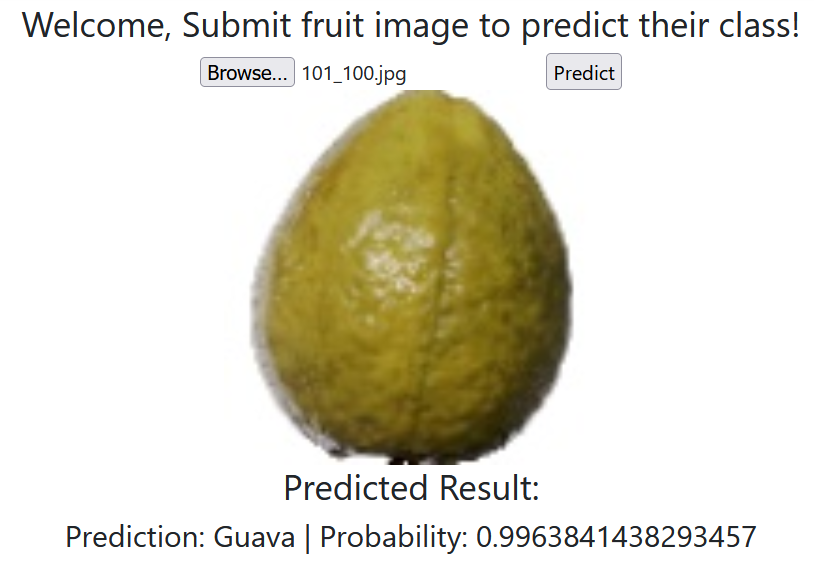
* + 1. Image Validation

However, if invalid file type is selected, system throws validation error and use is notified with error message.



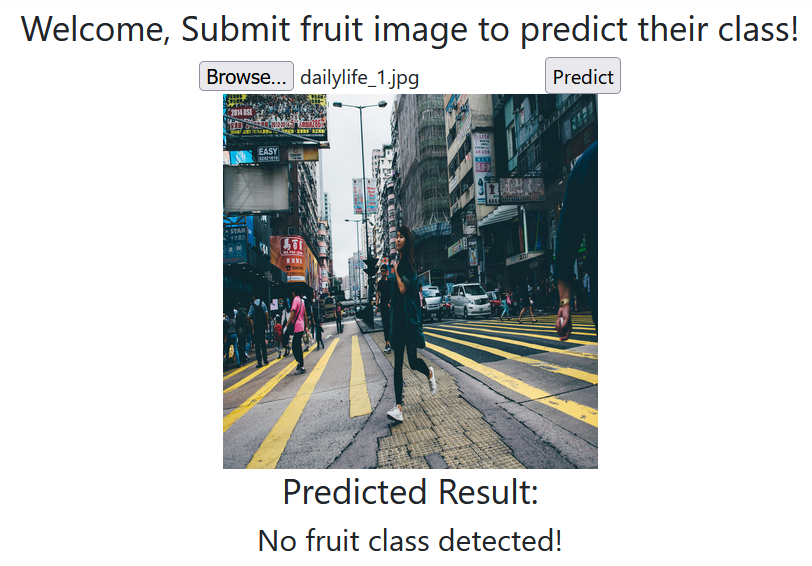
* + 1. Classification Result

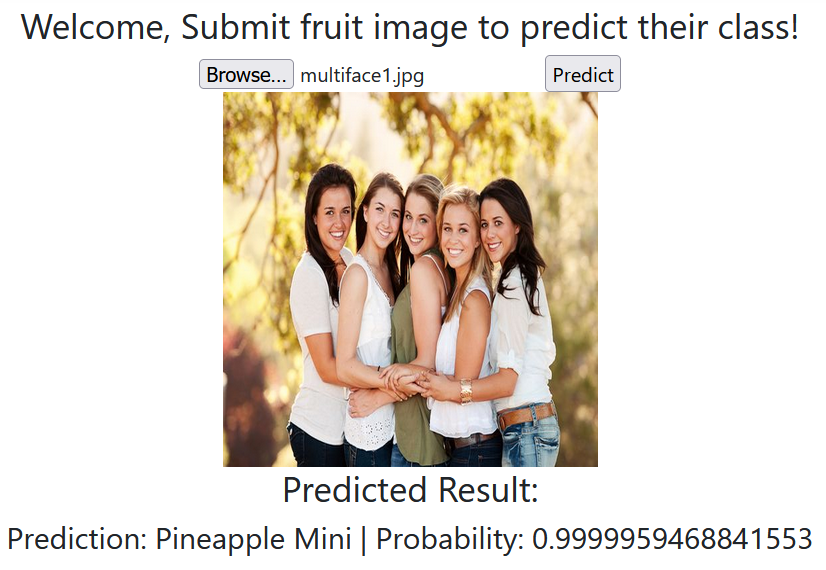
If the valid image is selected and predict button is clicked, system post the image to the API and classification result is then displayed to the client page.



1. Application Limitation
   1. Inconsistency with Image that does not contain fruit.

When we submit an image with new class (with no fruit) in them, application sometimes predicts as there are not fruit class, while sometimes it fails to do so.





1. Future Work

There are two major improvement areas. 1 model performance with unseen data and model compression.

* 1. In future I will be studying and improving the model to work better with unseen data as well as might implement one additional layer of classifier to first classify if image has fruit in it or not. Then, fruit category will only be called if there is fruit in the image.
  2. Another optimization would be compression of h5 file. This would help improving the API performance in terms of speed. I will be studying and experiment with **TFLiteConverter** to achieve this**.**

1. Jupyter Notebook



1. Jupyter Notebook Run Log

